(NASA-CR-166043) COST AND BENEFITS OPTIMIZATION MODEL FOR FAULT-TOLERANT AIRCRAFT ELECTRONIC SYSTEMS Final Report (Boeing Commercial Airplane Co.) 298 p HC A13/MF A01 CSCL 09C G3/33

N85-15993

Unclas 14901

NASA Contractor Report 166043

COST AND BENEFITS OPTIMIZATION MODEL FOR FAULT-TOLERANT AIRCRAFT ELECTRONIC SYSTEMS

FINAL REPORT

Boeing Commercial Airplane Company
P.O. Box 3707
Seattle, Washington 98124

Contract NAS1-16669
January 1983

National Aeronautics and
Space Administration

Langley Research Center
Hampton, Virginia 23665
AC 804 827-3966

NASA Contractor Report 166043

COST AND BENEFITS OPTIMIZATION MODEL FOR FAULT-TOLERANT AIRCRAFT ELECTRONIC SYSTEMS

FINAL REPORT

Boeing Commercial Airplane Company
P.O. Box 3707
Seattle, Washington 98124

Contract NAS1-16669
January 1983

National Aeronautics and
Space Administration

Langley Research Center
Hampton, Virginia 23665
AC 804 827-3966

FOREWORD

This report provides all the algorithms necessary to successfully develop a suite of computer programs for economic analysis of fault-tolerant systems.

The work was accomplished under NASA Technical Monitor George Finelli. Participants and their main areas of contribution were:

T. P. Enright	Program Manager
J. Rose	Sections 1, 2, and 6
A. N. Pozner	Section 5
R. C. Fairfield	Appendix A

Substantial portions of Section 4 and Appendixes B, C, D, and E were extracted from Reference 1. Special acknowledgement is given to M. Dockins and P. Stark who made significant contributions to the editing and review of the report. The project also is indebted to D. Ross of United Airlines for his assistance in defining maintenance practices of the real world.

CONTENTS

Page

Page

FIGURES

TABLES

1.0 SUMMARY

The use of active controls and integrated systems on future airplanes has a significant profit potential for the airlines and will require the development of highly reliable computer systems. Fault-tolerant systems (FTS) have the capability of achieving the high reliability required. The question of how much replication should be incorporated in a fault-tolerant system, and at what level it should be incorporated, is one of a number of questions for which economic analysis is required. This report sets out to both illuminate the problems associated with the exploitation of fault tolerance and to provide methods of optimizing and evaluating such systems from a design and airline operations standpoint.

A process of analysis has been developed called Cost Benefit Optimization in the form of algorithms for computer implementation. The algorithms form a model, shown in Figure 1, with three parts, capable of performing design cost optimization, design evaluation, and design economic analysis. From the few sample analyses that were possible without the benefit of complete computer programs, it is clear that replication as a method of implementing fault tolerance has two implications; on the one hand it improves short-term system reliability, on the other hand, it increases costs associated with adding levels of replication to the airplane.

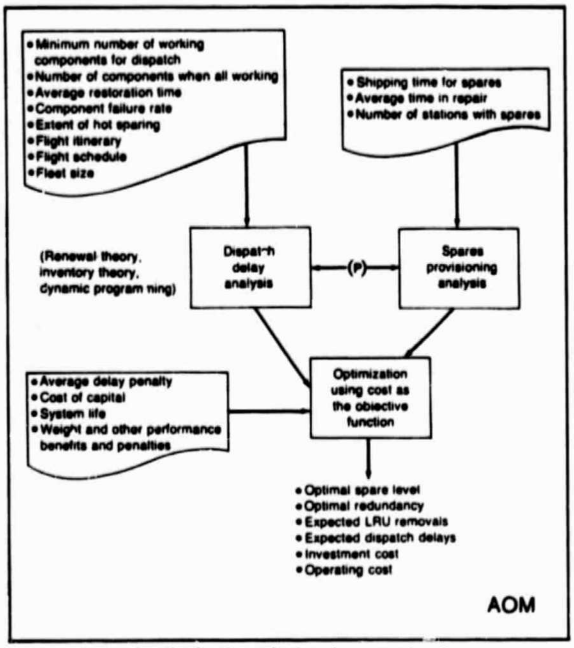
Both analytical and simulation techniques have been employed to ensure that all important economic aspects of fault-tolerant flight control systems (FTFCS) are accounted for. Both techniques have advantages and disadvantages. For instance, the analytical approach can be implemented to handle the large numbers of variable values involved in optimization, but it necessitates some simplifying assumptions. The simulation, on the other hand, provides a means of evaluating fault-tolerant systems in a comprehensive manner with few simplifying assumptions, but its optimization capabilities are very limited.

The potentially large computer run time requirements for the simulation and the difficulty in statistically analyzing simulation results were motivating factors in seeking analytical optimization methods.

The analytical optimization methods represent a significant advance in the ability to quantitatively analyze, understand, and design FTFCSs and other types of FTSs. All of the following factors can be analyzed for an airline environment by the analytical model:

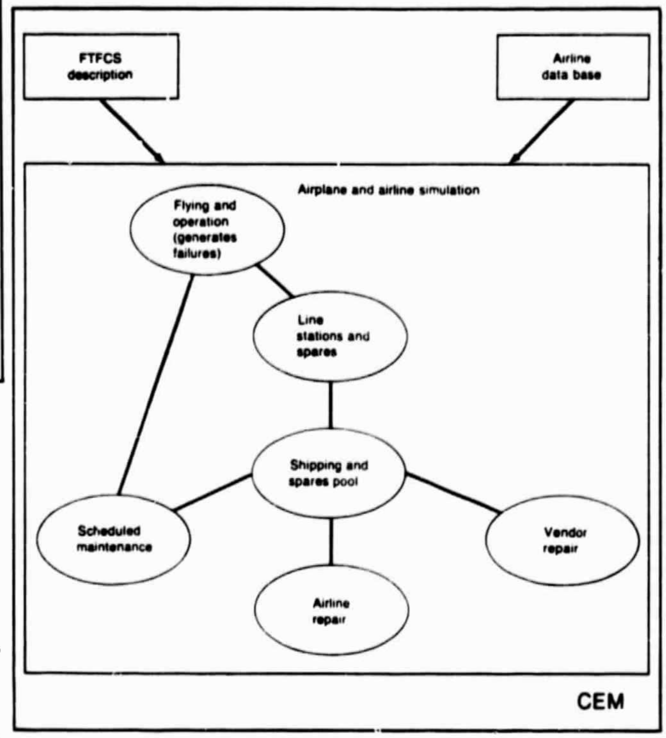
- Reliability of the components
- The ability to isolate failures correctly during maintenance
- Redundancy and packaging of the FTFCS
- Locations and quantities of spares
- The repair process and the effect of repair time of spares investment
- Dispatch delays and cancellations due to spares outage
- System availability
- Overall costs of investment and operation

ORIGINAL PAGE IS
OF POOR QUALITY



Analytical optimization method

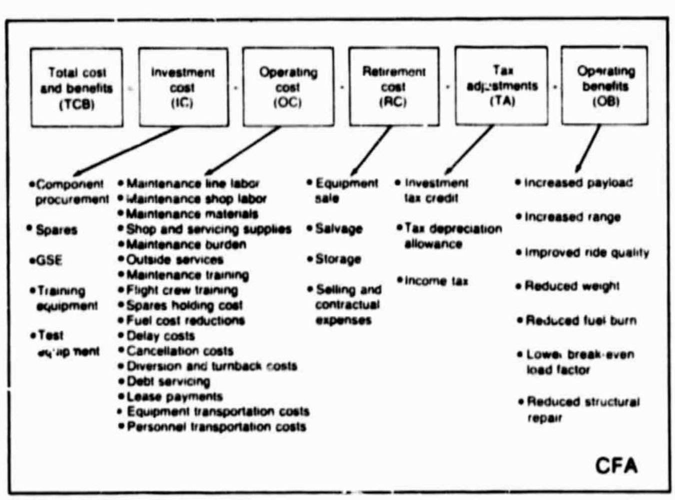
CEM and CFA are used to verify results of AOM and to provide a means of evaluating systems in specified optimal or nonoptimal conditions



Comprehensive evaluation model (Monte Carlo Simulation)

AOM assumes five activities can potentially reduce cost:

- Higher quality equipment (better reliability, packaging, fault isolation, durability, less space or weight)
- Additional redundancy
- Additional spares provisioned at line stations or depot
- More frequent scheduled maintenance
- More troubleshooting (to reduce repair time)



Cash flow analysis

Figure 1. CBOM Components, Structure, and Flow

The analytical optimization has modest computational requirements as a result of:

- Decomposition of larger problems into many smaller subproblems
- Representation of alternative designs and decisions by the same equations with different values of parameters
- Use of computationally efficient algorithms for the subproblems

The comprehensive evaluation model is based on Monte Carlo simulation and can be made to represent the real world in a simplified manner or in a comprehensive manner. Two important features unique to the simulation are the ability to produce statistics on the distribution of airplane delay times while waiting for repair and the distribution of times for repair of units within the repair shop.

The cash flow analysis consists of accounting algorithms to keep track of the costs and benefits of systems being analyzed. Most of the algorithms have been programmed previously and used extensively to make economic comparisons of design alternatives for the contractor's airplanes and for the NASA program on Integrated Application of Active Controls.

While the process of economic analysis using programs based on the developed algorithms can be accomplished in several different ways, it will usually consist of:

- Economic optimization of the replication, reliability, and packaging of components using algorithms in Section 4.0
- Evaluation of the functional performance of an optimized FTS in a realistic airline environment using algorithms in Section 5.0
- Determining the cash flows, payback, and return on investment (ROI) for an optimized system using algorithms in Section 6.0

The model also can be used for sensitivity analysis and to evaluate nonoptimal flight control systems, and there is the potential for developing parametric relationships using the simulation.

Significant accomplishments of the project include:

- A new mathematical representation of an airline environment using minimal input data (sec. 4.0)
- A new mathematical approach for modeling a k-out-of-n redundant systems (sec. 4.0)
- A heuristic optimization approach to jointly optimize functional stages and equipment management (sec. 4.4.2)
- A new application of a Markov model for optimizing scheduled maintenance intervals (app. C)
- Representation of functional and physical dependencies (secs. 4.1.5 and 5.2)

- An improved mathematical approach for computing mean time to system failure and upper/lower bounds of mean time to component removals (app. C)

- A new, computationally efficient algorithm for estimating probabilistic dispatch delay rates due to spares outages of each individual component as well as the system (sec. 4.0)

- Algorithms for estimating availability of non-dispatch-critical functions (app. C)

- Provision of a Monte Carlo simulation model specification that, when programmed, will evaluate fault-tolerant systems in a commercial airline environment (sec. 5.0)

- Development of a new method of simulating cancellations of flights using virtual airplanes

- Development of a method of allocating airplanes to flights within a simulation

Finally, for the complete project:

- A direct way of relating the characteristics of fault-tolerant systems to the cost and benefits they may produce in an airline environment

The prospect of a model with the analysis capabilities of the CBOM is exciting from both a design and airline logistic planning standpoint. We believe the CBOM is ready for the next steps; namely, coding, hands-on experience, refinement, and validation.

2.0 INTRODUCTION

2.1 SCOPE

The material covered by this document falls into two categories:

- That required for understanding the factors involved in economic assessment of fault-tolerant systems (FTS) and fault-tolerant flight control systems (FTFCS)

- That required to document algorithms for optimization and economic analysis of FTFCS

This section provides the background for understanding FTS and FTFCS analysis concepts. Algorithms to be used for analysis are included in Sections 4.0, 5.0, and 6.0. A Glossary is contained in Section 3.0.

While the objective of the project was the development of economic evaluation methods of FTFCSs, much of the work has been on the development of reliability and maintainability evaluation methods. The methods developed assume the analyst or designer will separately establish the safety bounds within which economic optimization is possible, and no attempt has been made to include evaluation methods for penalties or benefits associated with system safety. In the analysis method, benefits and penalties are established for functional or physical life-cycle changes in the state of the system. Functional changes are changes such as an increase in fuel burn and an inability to dispatch the airplane. Physical changes are replacements of components and assemblies of components. Physical change may or may not be accompanied by a loss of function depending on the way the system is implemented. Thus, an important concept of FTS economic analysis is the definition of both the physical characteristics of the system and the penalties from loss of functions of the system. A simplified example of such descriptions is shown in Figure 2. Figure 2a shows a primitive flight control system that might include authority limits so the airplane can be flown on one control surface. Lines in the diagram represent the paths for a successful system end-to-end function. Functional relationships are illustrated by Figure 2b where loss of a power source B may prevent an actuator C from working. The components also can have functional dependencies, as shown in Figure 2b, and physical dependencies with pairs of actuators packaged together, as shown in Figure 2c so that removal of C_2 results in C_1 being removed.

An essential difference between safety and economic analysis is that a designer, when engaged in performance or safety analysis, restricts his view to the systems of the aircraft. The system boundaries for safety do not generally extend beyond the aircraft. For economic analysis, however, the system boundaries need to include such things as operations and maintenance procedures, spares provisioning, and the airline route and schedule. Without this extension of boundary, the design of an economically optimal system is not possible. Its effect is to turn part of the design of FTSs into a difficult operations research problem in inventory management, queuing, scheduling, and renewal theory. Two approaches to solving the problem have been pursued: the first using an analytical method, and the second using simulation.

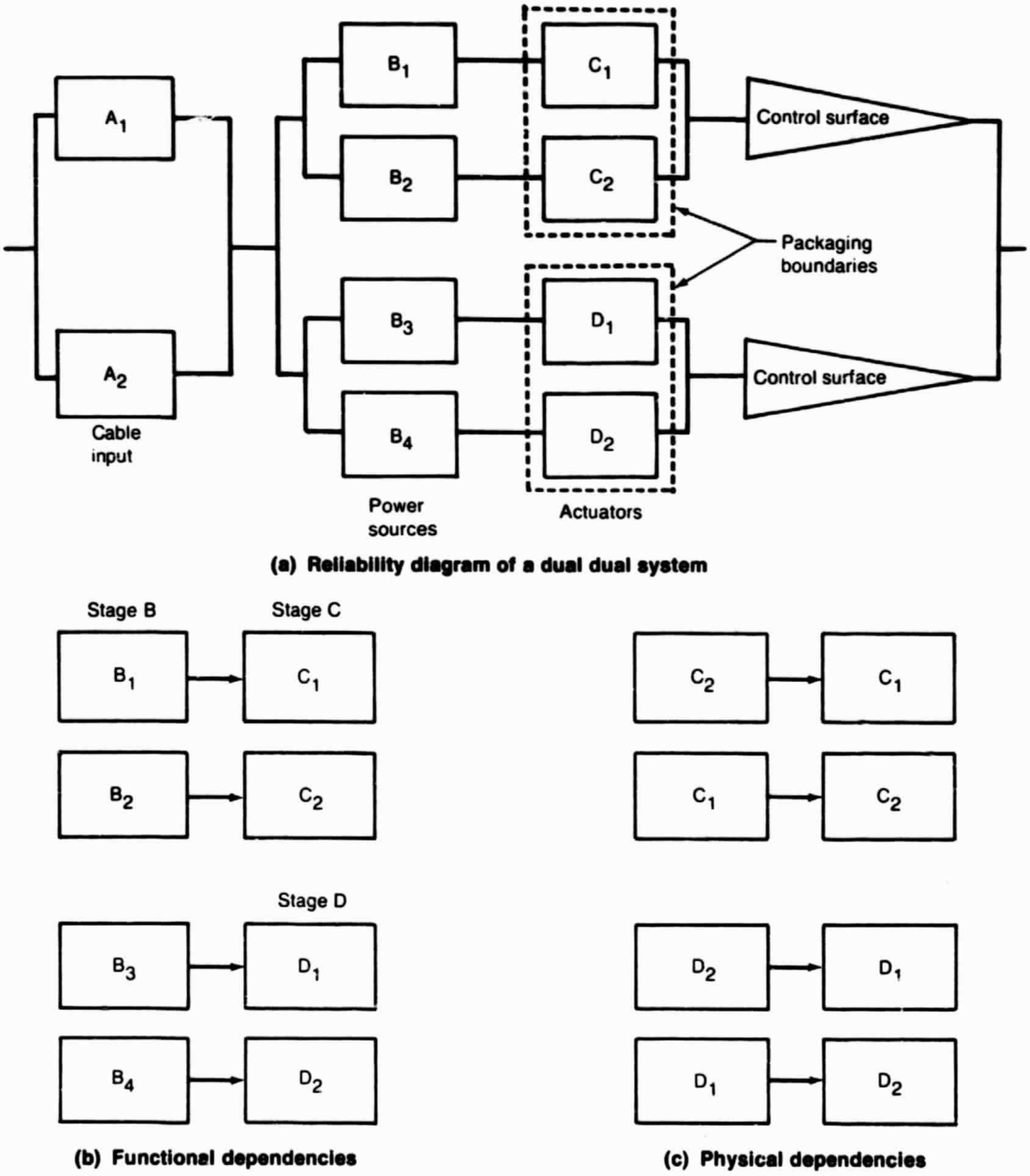


Figure 2. Reliability, Functional, and Physical Dependencies

The analytical method is economical to use and is well suited to optimization. For this reason it is called the Analytical Optimization Method (AOM). The simulation exactly models the stochastic relationships of both airlines and fault-tolerant systems, and for this reason is called the Comprehensive Evaluation Model (CEM) and will produce data for economic performance analysis that look like real world data. CEM data have the same problems of analysis as real world data because they also are samples from a complex stochastic process. The AOM and CEM are complementary in the sense that two values required as input to the AOM, namely the average cost of delays and the average time for LRU repair, can be obtained from the CEM. They also are complementary in the sense that the CEM uses the values of optimized variables, such as quantity and location of spares and levels of replication, as input. This is an iterative process that starts in either the AOM with an estimate of the average cost of delays and LRU repair time or in the CEM with a guess as to optimal replication, spares levels, and packaging arrangement. Output from the CEM also can be used to determine cash flows, return on investment (ROI), and payback point for a specified system and airline. This last analysis is made using a comprehensive method described in Section 6.0 called the cash flow analysis (CFA). Together, the AOM, CEM, and CFA shown in Figure 3 form a Cost Benefit Optimization Model (CBOM) capable of being used in parts or together for analysis of any commercial airplane design.

Some of the modeling concepts, problems, and issues that resulted in the algorithms defined in Sections 4.0, 5.0, and 6.0 are discussed in the remainder of this section.

2.2 AIRLINE OPERATING ENVIRONMENT

To cost optimize the design of FTSs, the designer must be concerned with two types of factors. The first type is under the designer's control and consists of levels of hardware and software replication, reliability, and the packaging of the components of the system. The second type is not under the designer's control and consists primarily of the airline route structure, fleet size, itinerary, and location of repair facilities. The term "decision variable" is used to describe the factors under the designer's control; the term "parameter," for the airline factors not under the designer's control. The CBOM provides the capability of optimizing all important decision variables including several different maintenance strategies. Thus, although airline parameters are not under the designer's control, they are within the designer's analytical purview and may be adjusted to produce the most cost-effective scenario for the FTS. Some discussion of the parameters is therfore important in understanding the model concept.

2.2.1 Route Structure

An important attribute of the route structure from an FTFCS standpoint is its topology. Figure 4 shows a number of typical routes. A hub-and-spoke route will obviously have spares and repair facilities located at the hub to reduce time required to ship spares to their place of need and to position spares at the most probable place of need. For a point-to-point route, a main base and spares depot in the center of the route will be logical for an operation that shuttles from end to end. For a circular spoke route, spares will be uniformly distributed at the maintenance stations over the route, except when replication is added to each

ORIGINAL PAGE IS
OF POOR QUALITY

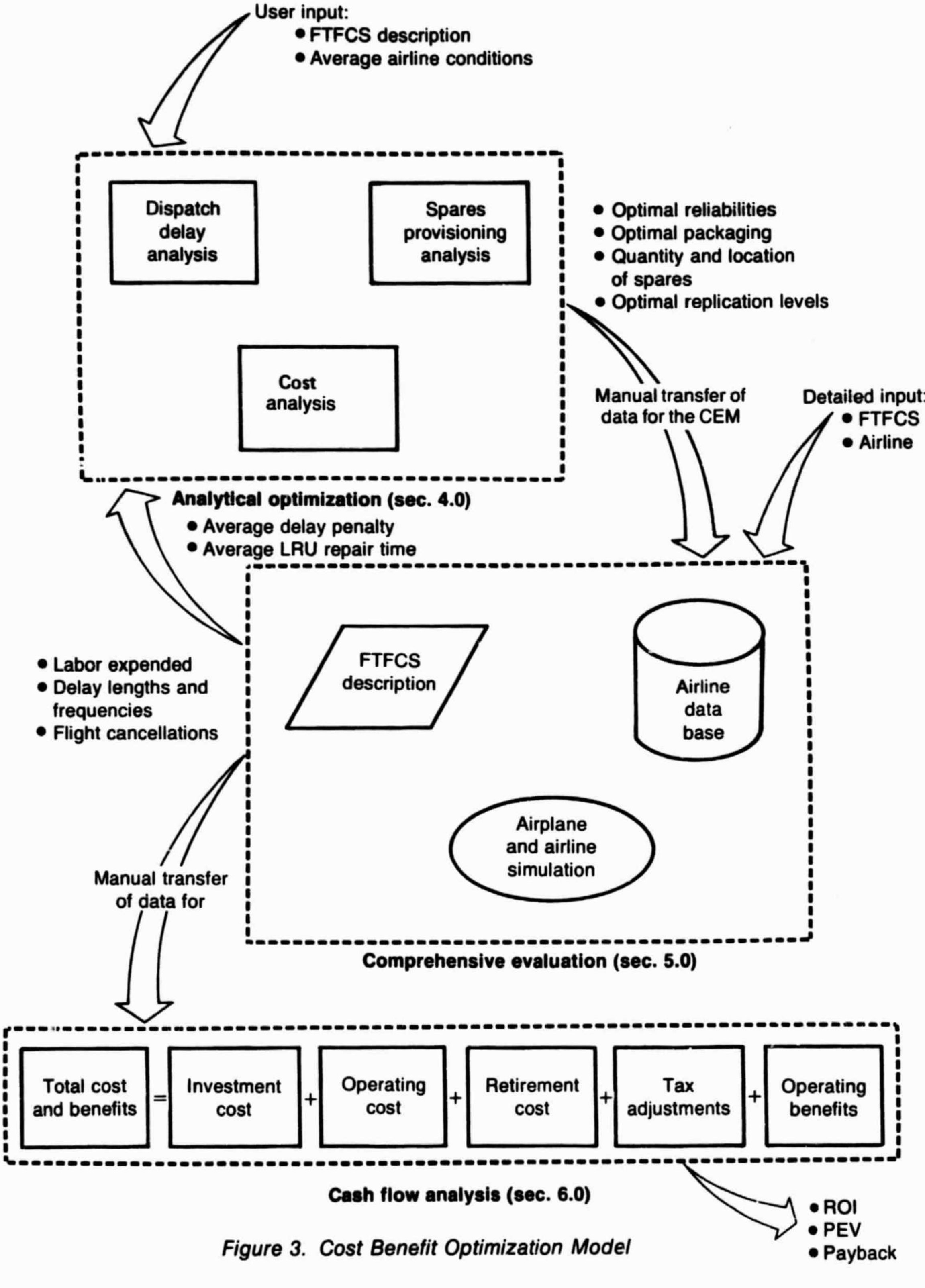


Figure 3. Cost Benefit Optimization Model

airplane, so that maintenance can be performed at a single base. Clearly, this type of design decision requires economic analysis. The question of how many spares should be provided at which stations on the route is not only a function of route topology and component failure rate but also the level of replication used in designing the FTS, the manner in which components are packaged, the cost of spares, and the cost of not having them when needed. The model developed provides a tool for answering the above questions and economically optimizing FTFCS design as well as the capability of evaluating the use of existing airline facilities that may not be optimal from an FTS standpoint. It also will provide a means for deciding if a single implementation of FTFCS is appropriate for all routes or if customized replication for particular routes is desirable. An FTFCS could conceivably be designed that uses add-on replication for dispatching flights to remote or overseas locations, thus providing a greater probability of return to home base without mandatory maintenance.

2.2.2 Fleet Size

There are two significant economic effects associated with the parameter of airline fleet size. The first concerns the ability to spread the nonrecurring costs of test equipment over a sufficiently large number of airplanes. The relative effect

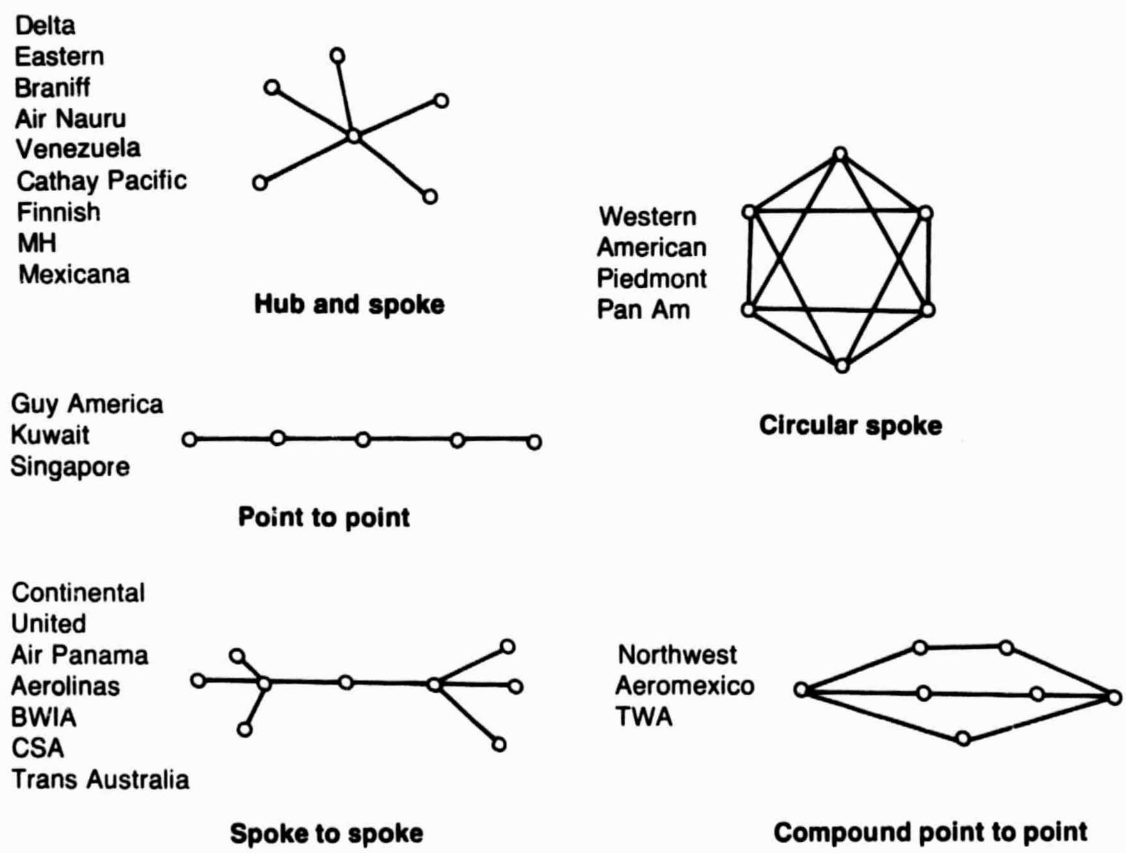


Figure 4. Route Topology

ORIGINAL PAGE IS
OF POOR QUALITY

can be seen from Figure 5 showing typical test equipment cost per airplane as a function of an airline's fleet size with the assumption that increasing fleet size does not necessitate additional test equipment. The second effect is the cost of spares per airplane. Spares cost is typically a function of the demand for a given item, the resupply time, and the protection level (or probability of no stock-out) that is adopted. Spares cost per airplane is also a function of fleet size. This effect, which can be seen in Figure 6 is significant for airline fleet sizes less than 40 or 50 airplanes. It thus appears that designing a system packaged in small units may significantly reduce spares cost for small airlines, but because of added weight, it might represent a cost penalty for large airlines that are not as significantly affected by a savings in spares. The CBOM has been developed to provide its user not only with the capability of evaluating the consequences of his selection of decision variables but also of evaluating the penalties such as those of a small fleet using equipment designed for a large fleet and vice versa.

2.2.3 Flight Schedule

The development of an airline's flight schedule is a complex process involving many schedulers who attempt to minimize the number of nonrevenue flights and to minimize losses from delays and cancellations by rescheduling and substituting airplanes. In addition, the amount of ground time available for maintenance has a significant effect on the number of delays that occur. Replicated components could be added to FTSs on airplanes that have tight ground time requirements and

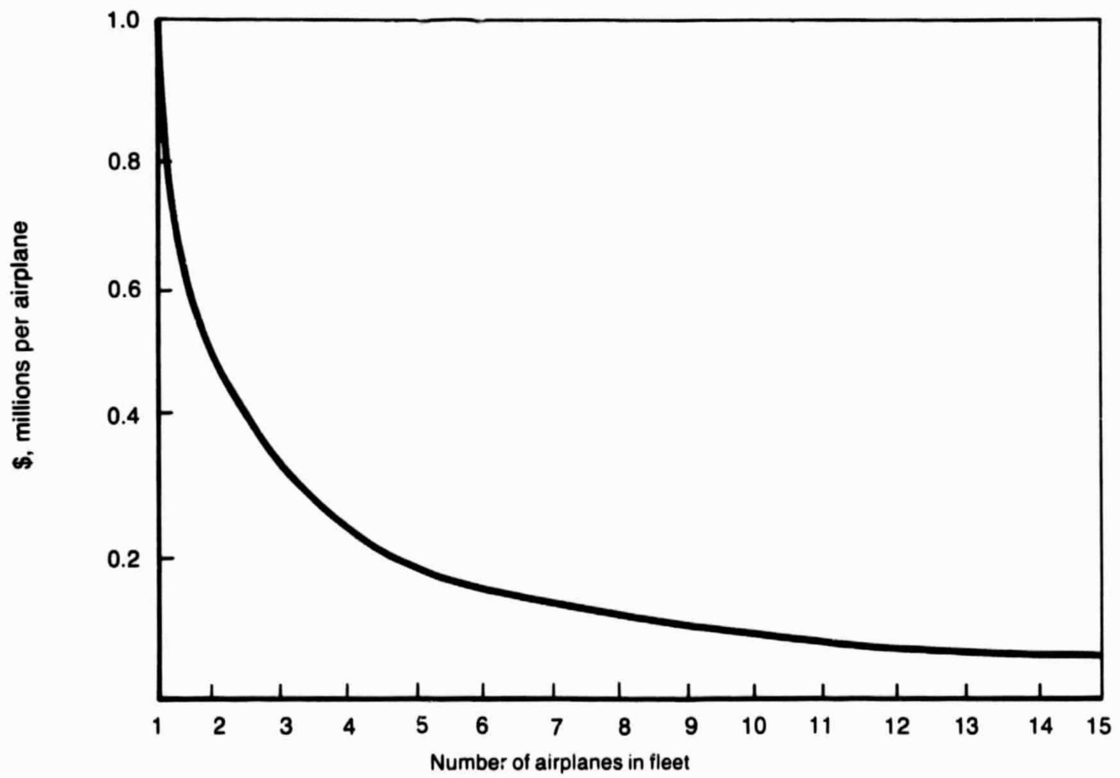


Figure 5. Effect of Fleet Size on Spread of Test Equipment Costs

the problem of finding an implementation of FTS that minimizes the cost of investment, maintenance, and delay is one of the objectives of the CBOM.

2.2.4 Maintenance Strategies

The CBOM will permit the user to examine the economic impact of different maintenance strategies such as:

- Periodically restoring the FTFCS to a fully operational state at a single base and maintaining it to a minimum dispatchable level between full restorations.
- Restoring the FTFCS to a fully operational state whenever time for maintenance is available, for example, overnight at a station normally stocking spares.
- Restoring the FTFCS to a fully operational state whenever it becomes nondispatchable.

Actual airline route structures and schedules for small, medium, and large fleets will be provided as a data base so that the input required of the model user is minimized.

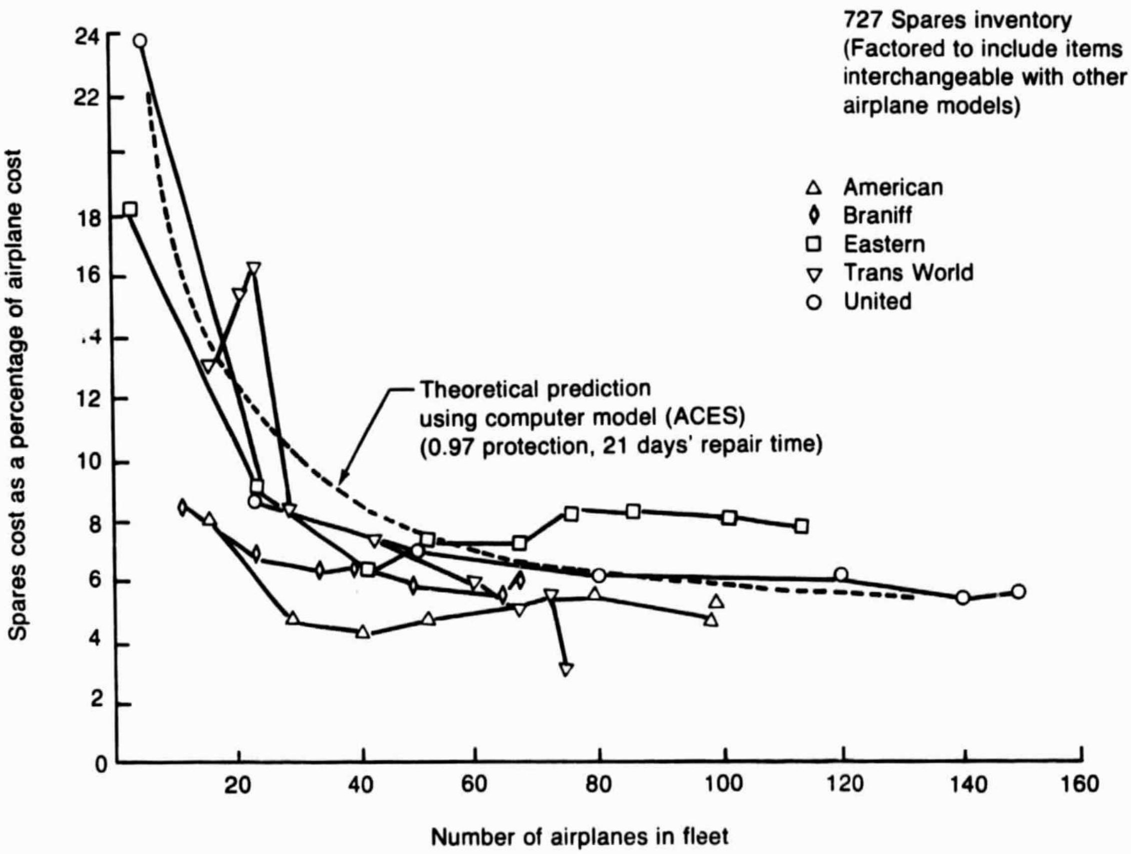


Figure 6. Effect of Fleet Size on Spares Cost

2.2.5 Standard Airlines

The philosophy of providing the model user with data bases for actual airlines is important for both optimization of conceptual design and detailed evaluation of specific FTFCS implementations. Boeing airplanes are distributed approximately as shown in Table 1.

Table 1. Fleet Size Distribution (1982)

FLEET SIZE	NUMBER OF OPERATORS	TOTAL AIRPLANES
1-5	228	570
6-10	31	230
11-25	54	970
26-50	15	570
> 50	15	1700

One of the potential uses of the AOM is to determine fleet sizes for which changes in FTFCS design occur. Based on such sensitivity analyses, an appropriate set of standard airline data bases can be chosen, thereby simplifying the problem of obtaining airline data each time an analysis is made.

2.3 FAULT-TOLERANT FLIGHT CONTROL SYSTEM MODELING

This section summarizes the more important aspects of fault tolerance as background for understanding the modeling of FTSs and the optimization and economic evaluation that follows. More detailed information on FTS architecture can be obtained from Reference 2 and Appendix A.

2.3.1 Principles of Fault Tolerance

Fault tolerance is the ability of a system to correctly perform specified external functions when differences between intended and actual functions exist within the system. Many different kinds of system architecture are possible. Figure 7 shows two frequently used fault-tolerant concepts, triple modular redundancy (TMR) and triplex-duplex-simplex (TDS). In TMR, two out of three components have output that agree when checked by a voter (V). Two failures are not acceptable, and generally the system will shut down when they occur. A TDS contains failure- or fault-detection capabilities at the component level, as well as voting. Internal fault detection in a TDS system tie-breaks two disagreeing outputs to the voter. Internal fault-detection methods are difficult to design, however, and do not provide comprehensive fault detection for digital systems in spite of word and bit counts, parity checks, and wraparound testing. Supplementing such techniques with signature analysis, Kalman filtering, and pattern recognition may use substantial portions of the system's computing capacity. Two concepts of fault-tolerant computers that may overcome the limitations of TMR systems are being developed by NASA and have heavily replicated and reconfigurable system modules capable of replacing failed members of a TMR triad. The NASA systems, called Software Implemented Fault Tolerance (SIFT) and Fault-Tolerant Multi-processor (FTMP), have architectures that permit use of six, seven, or more identical units, collec-

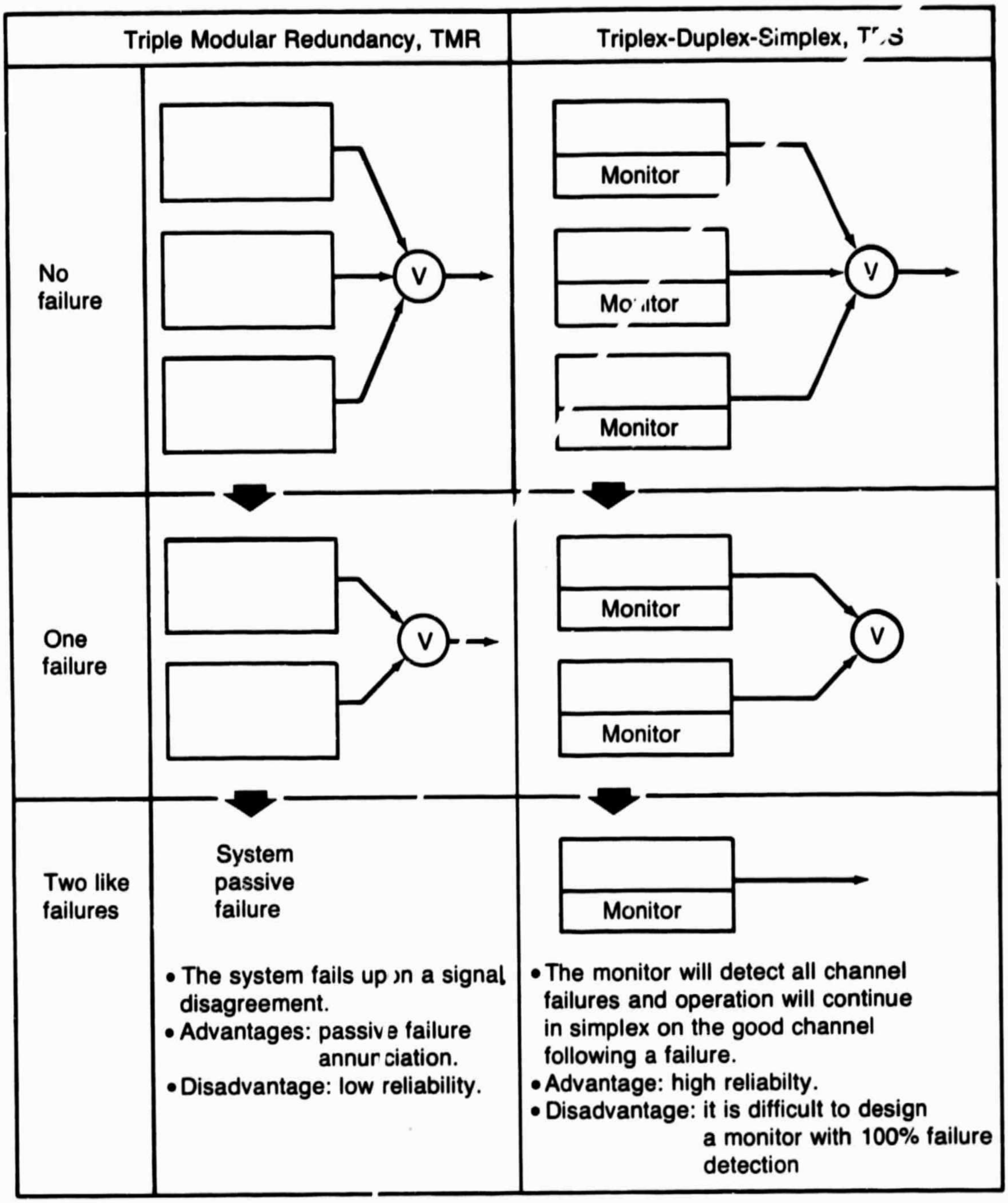


Figure 7 Fault-Tolerant System Concepts

tively called a stage, to achieve desired reliability. SIFT and FTMP have been used wherever possible for the examples of optimization and economic analysis in this report.

2.3.2 Fault-Tolerant System Attributes

The difference between the analyst's perception of an FTS from an economic analysis standpoint and of the same system for purposes of reliability analysis is based on the assumption that the reliability of any feasible commercial system will be designed to meet requirements such as those proposed by the Federal Aviation Administration (FAA), illustrated by Figure 8, Reference 3. It can be seen from Figure 8 that the FAA objective is a probability of catastrophe that is very small. In the case of SIFT and FTMP, NASA has adopted a requirement for a probability of catastrophe less than 10^{-9} at 10 hours. Though the cost of catastrophes might be large, they occur rarely compared to the large numbers of successful flights so that a catastrophe becomes an issue outside the area of economic optimization. The attributes of FTSs of importance from an economic standpoint are found at a much less detailed level of assembly than those for safety analysis. The system can be adequately modeled, from an economic standpoint, at a line replaceable unit (LRU) level or shop repairable unit (SRU) level, and several typical FTSs were reviewed to obtain an understanding of performance constraints that might limit economic optimization. The review has been documented in Appendix A, and some general characteristics of FTSs will be repeated here.

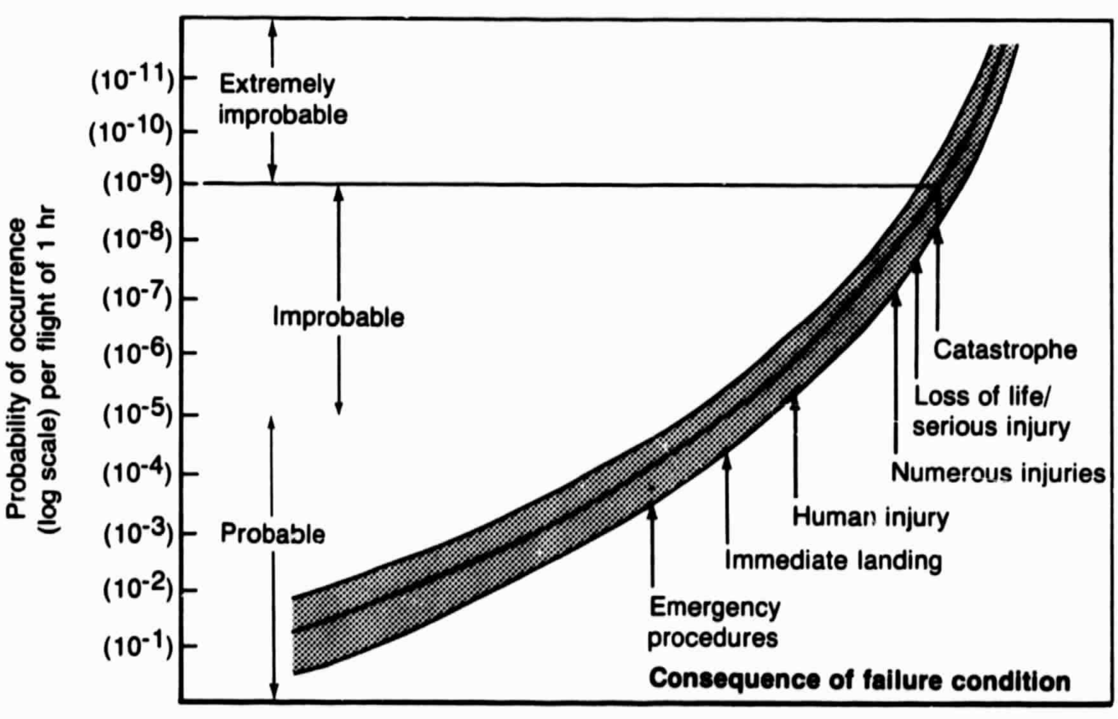


Figure 8. Relationship Between the Consequence of Failure and the Probability of Its Occurrence

2.3.2.1 Replication

Replication makes it possible to have situations similar to that shown in Figure 9 in which the system passes through a number of failure states immediately or ultimately requiring maintenance. The example of Figure 9 is typical of a SIFT-like system and in the illustration, degrades gracefully. Such a system might start with six usable processors, be dispatchable with five, and not experience significant performance or cumulative economic penalties until only four usable processors remain.

In the example, three types of maintenance are shown:

- A nondispatchable system is restored to a fully operational state.
- A nondispatchable system is restored to a dispatchable state.
- A system with a single failure is restored during periodic maintenance.

The AOM technique described in Section 4.0 enables such maintenance strategies to be compared and the optimal one for a given airline chosen.

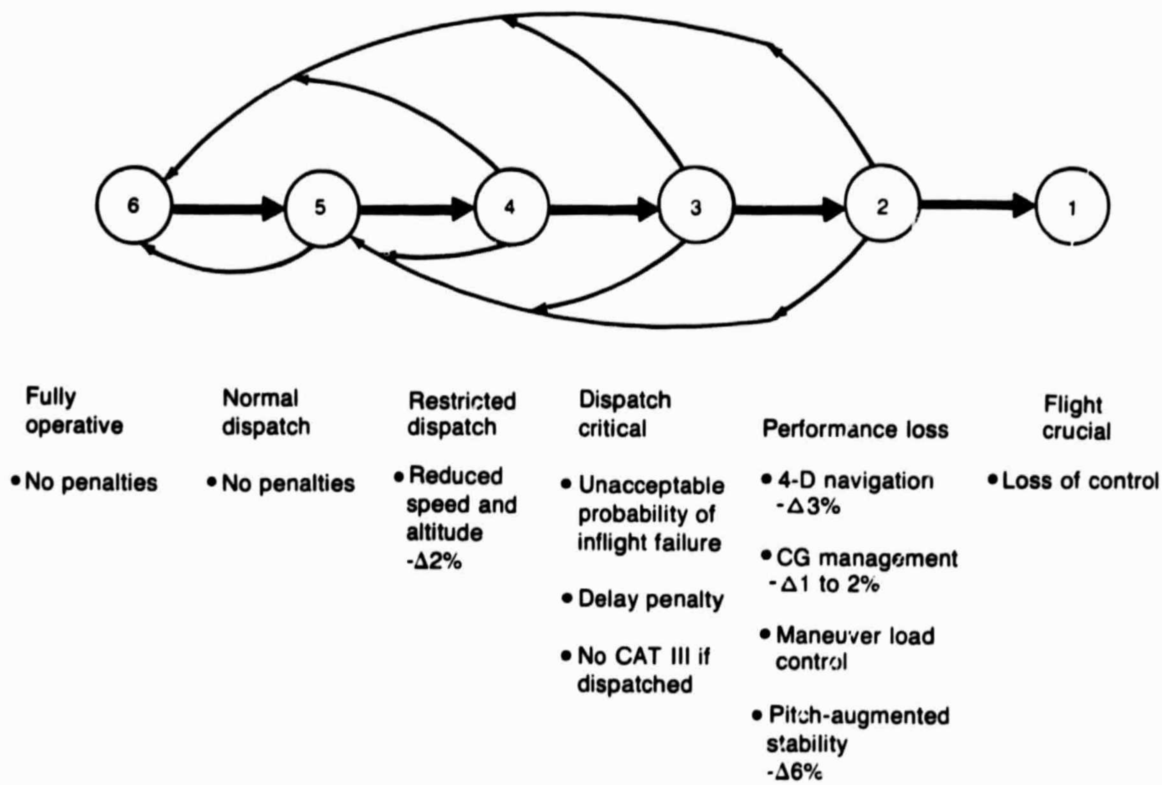


Figure 9. SIFT-Like System Degradation

2.3.2.2 System Recovery

Many fault-tolerant systems have a reconfiguration or self-recovery capability that provides the ability to perform predictably in the presence of faults. Recovery modes that may have economic consequences are:

- Full recovery, in which the system is restored by maintenance within a specified time to its full capability.
- Degraded recovery or graceful degradation, in which the system operates in a fault-free state with reduced capability.
- Masking, in which a fault is retained within a module of a system with no externally apparent change in the module's function.
- Fail-safe (or safe shutdown), in which warning is provided when system performance falls below a minimum acceptable criterion and is shut down to prevent consequential damage.

In all the above cases cost penalties can be determined and associated with the fault state of the system. Such cost penalties will generally be due to increased fuel burn because of flight restrictions, to delays waiting for mechanics or parts, to cancellation, and to flight diversions, except for the first case where system repair is the only cost.

One final comment on recovery. For purposes of safety analysis, the analyst may have to examine the probability of failure state changes in time intervals as short as a few millionths of a second; for economic analysis, sufficient resolution of system characteristics is possible in terms of hours or flights. Economic analysis is not easier as a problem, however.

2.3.2.3 System Test

Troubleshooting and testing account for a significant proportion of maintenance cost of contemporary avionic systems. Typically, about one-third of line maintenance labor is devoted to testing and fault isolation. Testing and fault isolation costs are as high as two-thirds of all labor in the case of avionic shop maintenance and in theory, fault-tolerant design concepts should significantly reduce costs for detecting and isolating failures. For instance, testing and fault isolation with an FTS can take place concurrently with processing, often under actual flight conditions. In addition, fault detection is an implicit part of the architectural design rather than an add-on and maintenance labor costs for system testing should be small.

Problems such as the testing of large-scale integration (LSI) circuits after replacement or repair have not been addressed other than allowing the CBOM user to include investment costs for logic and automatic testers.

2.3.3 Fault-Tolerant Applications

The gradual increase in airline acceptance of airplane safety-crucial avionic systems such as Category III auto-land is leading to investigation of new safety-

crucial functions such as active flight control systems. Active controls represent a logical extension of Category III auto-land systems. However, where Category III auto-land is required a few times per year for the short duration of descent and landing, active controls are required for the duration of every flight and must perform a wider range of control tasks than auto-land. The reliability implications of the significantly increased operation and risk exposure time are obvious. Substantial fuel economy is possible with active controls and may justify the design of fault-tolerant flight controls needed for safety of the concept. Active control surfaces for low-frequency directional stability augmentation or high-frequency structural load control may increase the payload of a typical commercial airliner several thousand pounds.

Clearly, an economic analysis is necessary to determine the investment in complexity that is supported by reduced weight and fuel burn. Such a study is typical of the use for which the CBOM is being developed.

Another potential use of fault tolerance is in the design of integrated systems, the driving force being both substantial weight reductions possible with integrated systems and the potential for reducing investment in microprocessors. For instance, while production costs per bit of memory of a given size (ref. 4) have typically dropped in a period of 5 or 6 years to a third of their former cost, the significant cost reduction has developed as a result of quantum jumps in memory size; a cost per bit of 65K memory is a tenth of the cost per bit of 1K memory. In the next decade, a move toward more integrated systems can be anticipated that will take advantage of the reduction in cost associated with increasing computional capacity in a given space. Many duplicated functions of nonintegrated contemporary systems will be eliminated.

2.4 PROCESS OF ANALYSIS AND THE CBOM

In its simplest terms, the process of analysis consists of:

- The assembly of data describing the FTFCS to be optimized or analyzed and the airline for which it is to be optimized or analyzed
- Exercising several computer programs to be developed from the algorithms of this document
- Interpreting the results from the computer programs

Sections 4.0, 5.0, and 6.0 show that the process is not a trivial one and cannot be accomplished without a computer. The analysis process can be made easier, however, by interactive programs to assist the analyst in selecting and organizing input data and in executing the several programs.

2.4.1 Analytical Optimization Method

The AOM is an analytical model of the stochastic processes involved in operating and maintaining FTSs. Because it is an analytical model, it can be used to compare alternatives that have small variable and parameter differences. Such comparisons, using cost as an objective function, are the basis for optimization. To make the AOM tractable, however, a simplified representation of the real world and the

ORIGINAL PAGE IS
OF POOR QUALITY

occasional use of bounds rather than exact determination has been employed. This makes the AOM economical to use and suitable as a means of optimization.

The AOM assumes that the significant costs, which separate competing FTFCS designs, are airplane delays, cancellations, and the capital cost of fault-tolerant equipment, both on the airplane and the ground. Therefore, there are two principal problems in the optimization. One is the determination of dispatch delay and cancellation costs, and the other is the determination of spares costs in terms of quantities of spares required and their optimal location (on the airplane, at a depot, or at line stations). Mathematically, the AOM consists of two sections linked by the probability (P) that a spare is available at the next stop. It is thus possible to determine the optimal value of P shown hypothetically in Figure 10 by using P in both the dispatch delay and the spares provisioning analysis.

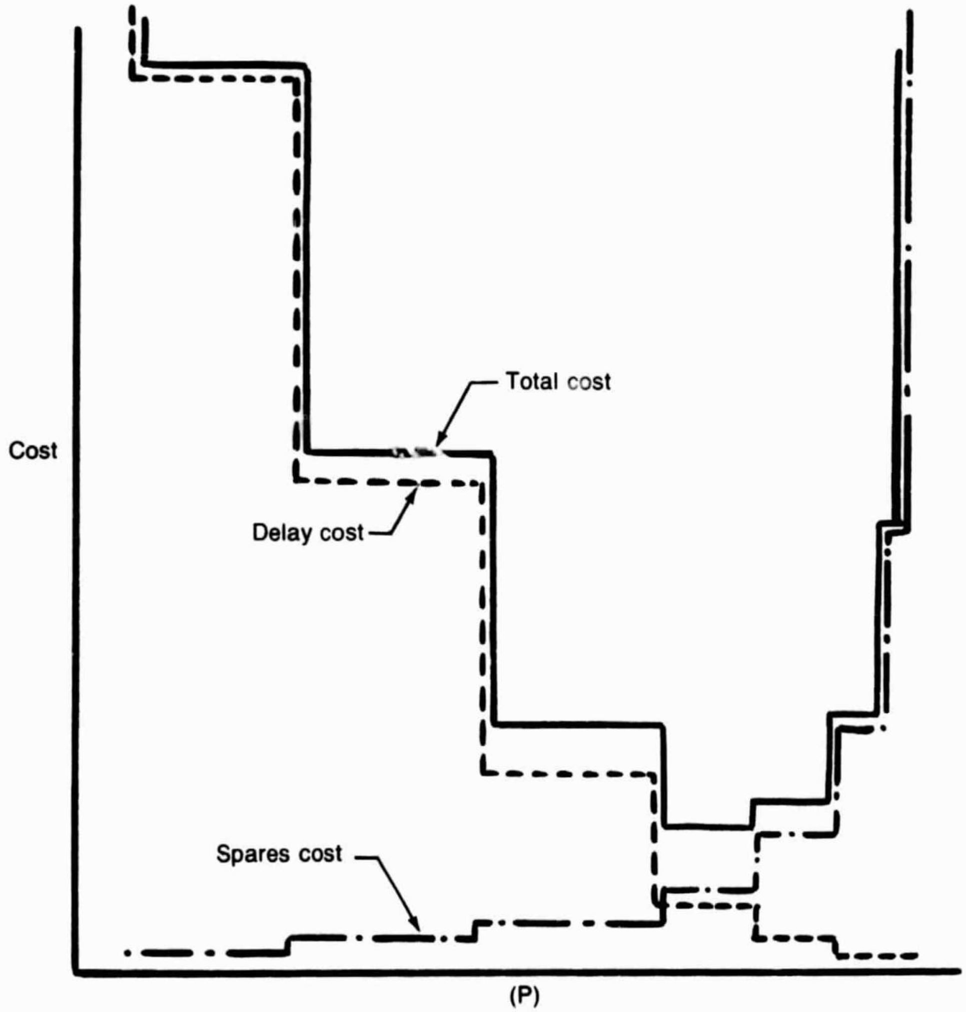


Figure 10. Example of the Minimum Total Cost of Spares and Delays as a Function of the Protection Level (P)

ORIGINAL PAGE IS
OF POOR QUALITY

The relatively few inputs required for optimization are shown in Figure 11. Some of the inputs could be supplied as default values by providing standard airline evaluation models for design study purposes.

2.4.2 Comprehensive Evaluation Model

The CEM has been designed to accurately represent the functions of an airline that affect or are affected by the operation of an FTFCS. These functions are shown in Figure 12 and are expanded to a more detailed level in Section 5.0. In contrast to the AOM, the CEM requires a substantial and detailed description of both the FTFCS and the airline. Much of the data describing the airline can be made available in a default data base; however, it still will probably take the analyst several days to prepare input describing the FTFCS.

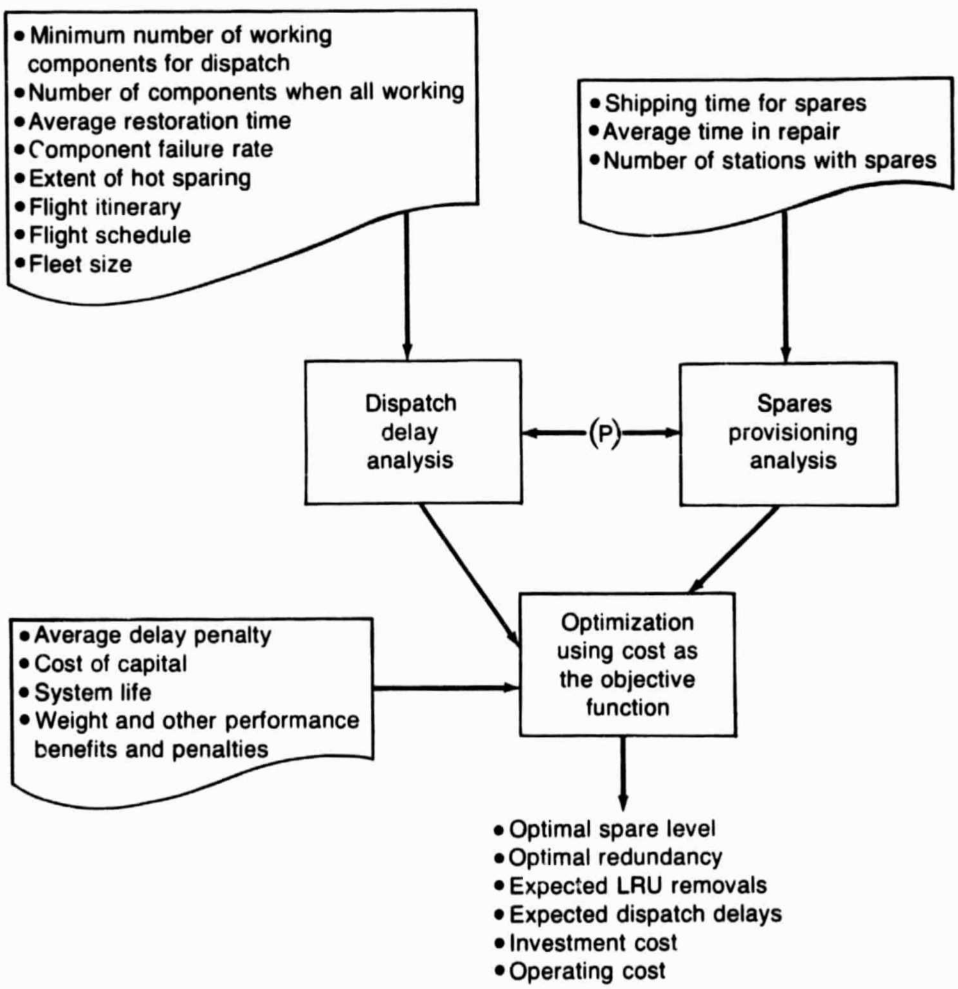


Figure 11. Analytical Optimization Method

The CEM has been designed as a comprehensive model of the real world without the necessity of establishing the analytical relationships between different parameters and variables. The results from the CEM are random variables, however, and therefore are subject to statistical interpretation. This makes the CEM expensive to use to determine the effects of small changes in variables or parameters. For example, approximately 10^8 flights must be simulated so there is a 98% probability (or confidence) that a delay rate does not differ from a simulated rate of 1 in 10 000 flights by more than $\pm 2\%$. To simulate 10^8 flights may require a run time of several hours on a mainframe computer such as a Cyber 175, and the cases to be evaluated using comprehensive simulation, therefore, should be carefully chosen.

It would be even more expensive to use the CEM to determine if any of the variables or parameters were dominant, making its use inappropriate for optimization. The appropriate uses for the CEM are to—

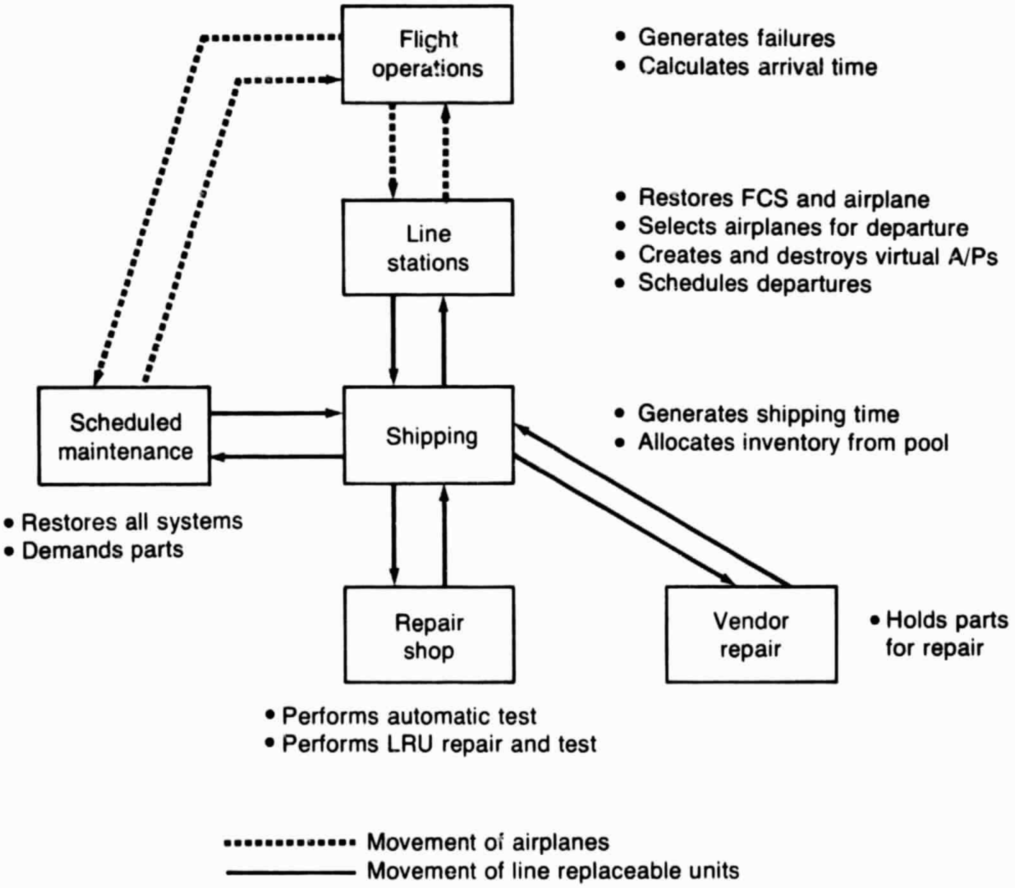


Figure 12. Comprehensive Evaluation Model (Monte Carlo Simulation)

- Determine the frequency of delays and cancellations and the duration of delays that occur with a specific route structure and itinerary. This information, when converted to an average delay penalty with the algorithms of Section 6.2.12 and 6.2.13, can be used as input to the AOM.

- Determine the resources required to maintain the FTFCS in terms of labor and test equipment using optional spares determined from the AOM or some nonoptional policy desired by an airline.

- Establish the adequacy of specified repair shop, test, and repair provisions.

- Determine the adequacy of assumptions and approximations used in the AOM.

In the CEM, delays and cancellations arise from parts outages, labor shortages, tight ground times, and stochastic behavior of the system. The FTFCS contends for labor resources used by other avionics on both the common and different airplane models.

In addition to delay and cancellation statistics, the CEM yields the pre- and postflight statistics on FTFCS failure states to enable flight performance benefits and penalties to be assessed (sec. 6.2.11).

To use the CEM, the analyst first constructs or modifies seven input data bases. After verification checks, the input data bases are used to drive the CEM simulation, and outputs are generated. Section 5.0 provides a detailed breakdown of input, output, and processing for each part of the simulation. Provision has been made to terminate the simulation when statistics of interest have been determined with a confidence level specified by the model user.

2.4.3 Cash Flow Analysis

Two economic analysis routines have been provided. One is to be employed as part of the AOM and uses only the variable costs necessary for optimization. The other will be used for determining the economic feasibility of specific FTFCS configurations and produces figures of merit in terms of return on investment, present value cash flows, and payback point. This latter routine is based on the Boeing computer program Airline Cost Estimating System (ACES). Algorithms required for the CBOM have been extracted from ACES and are included in Section 6.0. Some new algorithms covering such things as lease payments, debt servicing, and equipment transportation cost have been added. Costs and benefits that should be considered in economic analyses of different FTFCSs are shown in Figure 13.

2.5 RESOURCES FOR IMPLEMENTATION

In conclusion, some speculation is required on the resources required to implement the model and on the skills required for its use. Programming the algorithms of Sections 4.0 and 5.0 should be a relatively straightforward process because the model lends itself to a modular construction. Most of the algorithms in Section 6.0 have been previously programmed. Some development and experimentation will undoubtedly be necessary before a user friendly model has been successfully completed, and the flexibility to change the software is an important part of the program design and coding.

2.5.1 Program Implementation

The CEM, being a Monte Carlo simulation of intermediate size, is most economically and flexibly programmed in a high-level simulation language. SIMSCRIPT is the most appropriate choice of languages in terms of programming labor, execution cost, programmer availability, ease of documentation, and ease of change. The simulation can be run on NASA's CDC-6000, Cyber 175, or NASA Airlab's VAX computers, all of which can support a SIMSCRIPT compiler.

The appropriate language for programming the AOM and CFA is Fortran because of the advantages of code portability and existing CFA subroutines. No problems are anticipated in implementing either the AOM or CFA on NASA's CDC-6000 series computer or VAX computers in NASA's Airlab.

With good input file design, much of the input to the AOM and CEM will be common, and output files can be created so that output from one CBOM module becomes input to the next module. Supplemented input data will be provided interactively by the analyst before program module execution. The analyst will intervene between the AOM, CEM, and CFA economic analysis to determine the desirability of continuing an analysis after inspection of results and to provide additional input data required to continue.

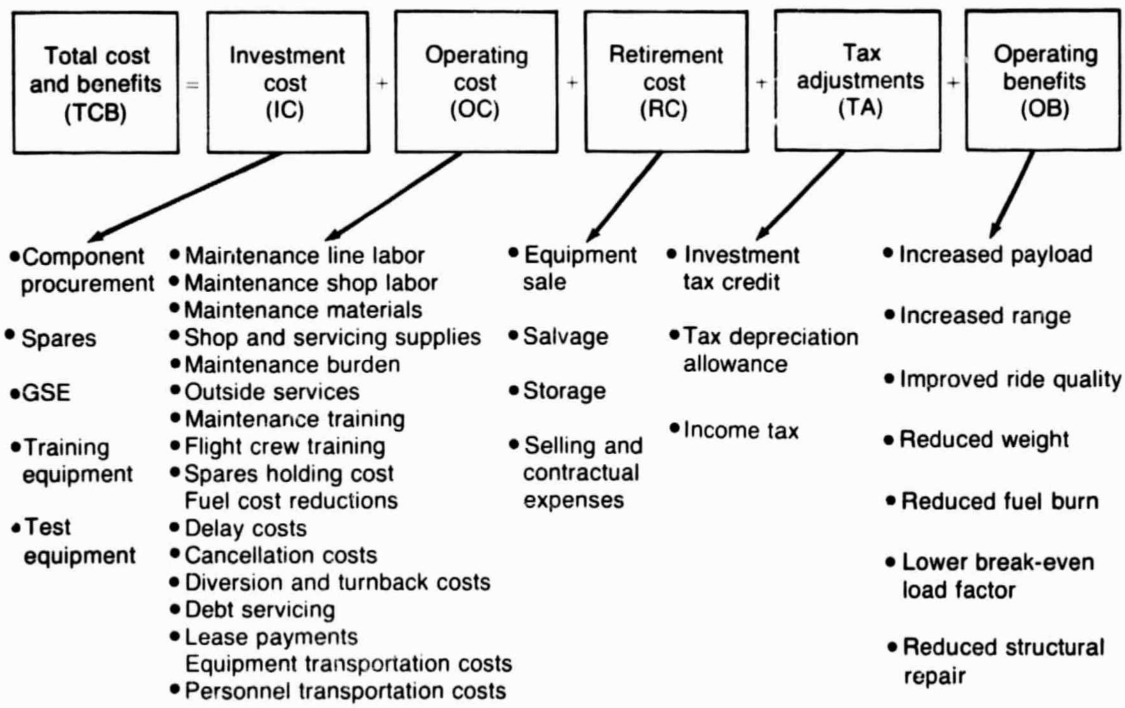


Figure 13. Costs and Benefits

The run time per case for the AOM and CFA will require considerably less than those resource limitations of Langley Research Center's Specification for Computer Programming (revised October 7, 1976). Runs of the CEM will sometimes be required in excess of the NASA 15-minute run time limit, and a concession will be required to make long batch runs during off-peak periods.

The CBOM has been designed using a modular approach, thus ensuring functional independence within the program modules. Each module will be physically separate, allowing flexibility in level of detail and modification within the module with little impact on the other modules. No modification to the CBOM code will be required to initialize or modify a model attribute, system description variable, or FTFCS configuration characteristics. This requires a very large number of data elements to be input to the program in the case of the CEM. In order for the analyst to be assured that the data being input to the program is complete and correct, a module to check the input data bases for errors is necessary. This module will ensure that the CEM does not fail because of incomplete data nor generate inaccurate results because of incorrect data.

2.5.2 Resources for Analysis

Resources required to perform an analysis in terms of labor and skill level are contingent on which of two policies for analysis are chosen. The policies are:

- Use a specialist to perform cost and benefit analyses on behalf of the systems designer
- Train the systems designer to perform the analysis.

The pros and cons of these two policies are:

	PRO	CON
CBOM specialist as the analyst	• No major training in CBOM is required. • Simpler user manuals. • Analysis quickly becomes routine and efficient. • Familiarity with data sources makes development of input easy. • Familiarity with the model makes interpretation of results and debugging easier.	• Each time an analysis is made, the system architecture and details must be understood. • There can be a communications problem with the system designer. • There is little understanding of the performance implications of architectural changes.

	PRO	CON
FTFCS designer as the analyst	• Intimate familiarity with the design of the FTFCS.	• Inhibited against changing his own design.
	• Knowledge of what may be treated as variable within FTFCS performance limits.	• Considerable programming effort is required to make the CBOM friendly for an infrequent user.
	• Experience in developing optimal design policies that may influence future design.	• There is little understanding of the cost drivers in airline operation.

Both policies have been tried by the contractor for techniques similar to the CBOM, and under conditions of limited budget, the use of CBOM specialists is the expedient policy.

Prerequisites for a CBOM specialist are:

- A background in avionics engineering
- A good knowledge of system analysis and operations research
- An ability to work closely with the FTFCS designer

2.6 FUTURE DEVELOPMENT

This section summarizes work that might usefully follow the verification of each of the CBOM programs.

2.6.1 AOM Developments

Seven enhancements to the AOM are detailed in Appendix C. They include enhancements to make the AOM easy to use and include algorithms that improve the accuracy of the optimization process.

2.6.2 CEM Developments

The CEM has been designed with no intentional departure from an accurate representation of the real world. In consequence, there undoubtedly will be some second-order features of the model that can be eliminated by using the completed CEM experimentally. Hands-on experience with the CEM is required before simplifications or creation of a data base of "standard" airline data can be accomplished.

Hands-on experience also will be useful in determining the appropriate experimental design and method of analysis to be used. If the FTS is independent of other systems of the airplane as far as demands on repair resources are concerned, then results can be obtained from the simulation using a technique to establish confidence limits on results such as that of Section 5.9.2. For FTSs where such convenient independence does not exist, because of contention for common resources for example, the method of analysis is not as clear. Several relevant papers on simulation analysis have been published at IEEE simulation conferences that warrant future investigation if the assumption of independence is shown by the CEM to be invalid.

2.6.3 CFA Developments

Four cost-estimating relationships detailed in Section 6.0 will be improved in accuracy if use of the AOM shows them to be important. The relationships are those for delays, cancellations, turnbacks, and spares holding cost.

2.6.4 Validation and Testing

For validating and testing the CBOM, several levels of analysis are desirable. The first level compares the predicted optimal configuration with some existing fault-tolerant equipment for several airlines. Although designed to work for new systems for which there is no historical airline data, the CBOM also applies to existing systems. There should be no major difference between the model's optimum and the redundancy level, location of spares, and number of spares of existing fault-tolerant systems. In using historical data for external validation, allowance should be made for historical policies that are not optimal under today's conditions. For example, a historical inventory policy based on zero parts shortages may have been optimal when holding costs were low but may not be optimal now. Thus, historical inventory levels might provide an upper bound for optimal levels specified by the model.

A second level of validation of the probability calculations of dispatch delay and availability is also required for the AOM. Also, the dispatch portion of the AOM is essentially a modified reliability model and thus, an appropriate comparison with results of an FTFCS design reliability model (such as CARE III) is a logical method of validation. This kind of validation is illustrated in Section 4.5.

A third level of testing that can be loosely termed validation is to conduct a series of sensitivity analyses to determine the sensitivity and the optimality of the default values for parameters and policies. For example, to allow the designer to focus on the design, the computer will automatically optimize the operation and maintenance. In this context, a default maintenance policy will be selected where the repair of any failed LRU is considered deferrable unless it brings the number below the minimum required for dispatch; it will be replaced at the next overnight stop at which a replacement LRU is stored and available. Use of the model will determine under what conditions the default policy is optimal and whether or not the designer needs to be concerned with alternative maintenance policies.

In the case of the CEM, additional validation will take place using resources of several airlines to ensure that the queues for resources, airplane substitutions, and delays, that occur in the simulation match those of the real world.

NOTE: The expressed or implied use of commercial products or names of manufacturers in this report does not constitute official endorsement of such products or manufacturers by the National Aeronautics and Space Administration or by the contractor.

3.0 GLOSSARY AND ACRONYMS

3.1 GLOSSARY

ACTIVE CONTROLS

The use of control systems to improve an airplane's performance (reduced drag and reduced weight) through the incorporation of feed-forward or feed-back control systems that augment the airframe stability and control or reduce loads on the airframe.

ADDRESS

A unique code for recording the location of a component in a functional stage or physical package.

ADDRESS AFFILIATION

The physical linkage of an FCS component with the package that contains it and the logical linkage of the component with the functional stage to which it belongs; thus, a particular address is affiliated with both a package and a stage.

ALGORITHM

A well-defined set of processes or rules for the solution of a problem in a finite number of steps.

AIRCRAFT ON GROUND (AOG)

The hightest priority designation to process a requirement for spare parts or maintenance action. AOG indicates that an airplane is unable to continue or be returned to revenue service until the appropriate action is taken.

AVIONICS

Electrical and electronic devices used in aviation.

BLOCK TIME, BLOCK-TO-BLOCK TIME

The time between an airplane's moving from the gate and stopping at the next gate.

CAUSE ADDRESS

The address of a component whose failure causes failures or nullifications elsewhere.

COMPONENT

A basic building block used to represent the system being modeled that has known characteristics including cost and failure distribution. Each component belongs to both a functional stage and a physical package.

COMPONENT HIERARCHY

A means of distinguishing the four component levels treated by the CEM:

Level 1:	LRUs that contain no line replaceable subunits
Level 2:	LRUs that contain LRUs, SRUs, or both
Level 3:	LRUs contained within a level 2 LRU
Level 4:	SRUs contained with a level 2 LRU

CONDITION MONITORING (CM)

A maintenance philosophy that uses an appropriate means of data collection and analysis by which a carrier obtains information from the whole population of a system. Condition monitoring includes the elements shown in Figure 14.

CORRECTNESS

The complete agreement of performance with specifications.

CORRECTNESS PROOF

A mathematical proof that a given computer program's performance is consistent with its performance specifications under all specified operating conditions.

COVERAGE

The conditional probability that, given a failure has occurred, the system will continue to function as specified.

CUMULATIVE CASH FLOW

The sum of annual revenues or expenditures over a specified number of years.

DEFAULT

Data used by a computer program that may be changed in an input data stream at the discretion of the program user.

DEGRADATION

A reduction in capability due to the presence of faults.

DEPENDENT EVENTS

Failures or nullifications or both caused by failures elsewhere.

DESTINATION OF INTEREST

The flight leg to which an airplane must be assigned.

DRAW

A process of obtaining a value of a random variable having a defined probability distribution.

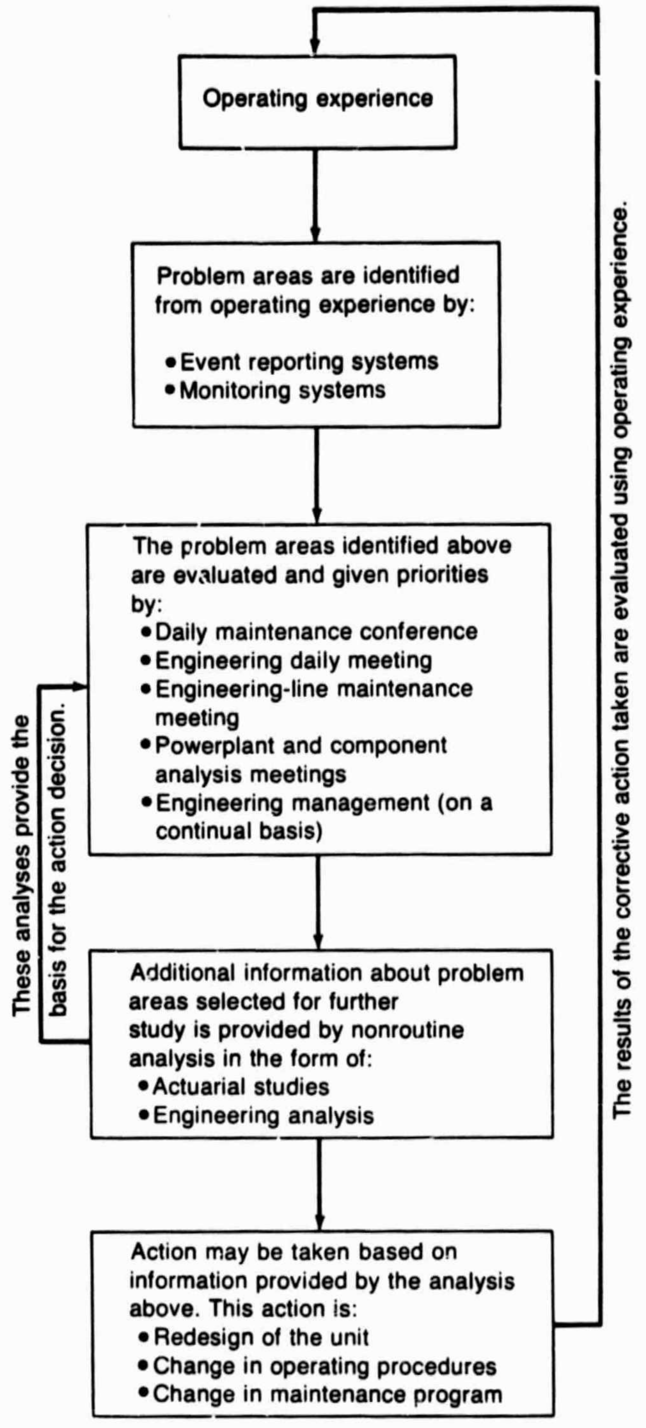


Figure 14. Maintenance Program Condition Monitoring

EFFECT ADDRESS

The address of a component whose functions become unavailable as a consequence of a failure elsewhere.

ERROR

The measurable or observable difference between intended and actual function or output.

EVENT NOTICE

A notice used to record the time at which an event is to be simulated so that events take place in proper time sequence.

EXISTING AIRPLANE

The airplane first assigned to a flight.

EXPENDABLE OR THROWAWAY

Items for which no authorized repair procedure exists and whose cost of repair would normally exceed that of replacement. Such items are also called "throw-aways."

FAILURE

An event that causes a system or subsystem to make a transition from a good state of generating no errors to a fault state in which errors are generated.

FAILURE, CRITICAL

A failure that results in a potential safety hazard that can be averted by appropriate crew actions.

FAILURE, CRUCIAL

A failure that may cause loss of life or total loss of the airplane.

FAILURE, DEPENDENT

A component failure caused by the failure of another component.

FAILURE MODE

A distinct physical manifestation of failure.

FAILURE SYNDROME

A distinct pattern of failure symptoms.

FAILURE VISIBILITY TIME

The airplane check at which a failure becomes visible for line maintenance purposes.

FAULT

A state of a system or subsystem in which a condition or set of conditions exists that may cause errors to occur and, in turn, may cause a failure in a higher level system.

FAULT DETECTION

With a system in the operating mode, the act or process of determining the presence of a fault (with or without isolating it) by observing an error.

FAULT DIAGNOSIS

Determining the causes of a fault in order to select proper reconfiguration actions.

FAULT, HARDWARE

A physical condition that may cause errors.

FAULT, INTERMITTENT

A fault where an error is repeated either periodically or aperiodically.

FAULT ISOLATION

Locating a condition that may cause errors.

FAULT, LATENT

A fault that is not generating an error.

FAULT MASKING

Overriding the errors caused by faults (one of several ways of achieving fault tolerance).

FAULT, SPECIFICATION

Software, firmware, or hardware functions that may cause errors as a result of incorrect documentation.

FAULT, SOFTWARE

Program code (bug) that may cause errors.

FAULT TOLERANCE

The ability to perform as specified in the presence of faults.

FAULT, TRANSIENT

A fault causing a one-time occurrence of errors.

FIRMWARE

A device that responds to computer instructions that are a part of that device's hardware.

FIRST AND ADDITIONAL PIECE

When an LRU requires restoration of N identical components, there is one FIRST piece and N-1 ADDITIONAL pieces.

FLAG

A program indication that an event has or has not taken place.

FLIGHT ITINERARY

A listing of all flights to be performed by a fleet of airplanes built up from flight tours.

FLIGHT LEG

A single flight from and to one city or between a city pair with details of the scheduled departure and arrival time and date.

FLIGHT LINEUP

An allocation of airplanes scheduled to cover the flight itinerary.

FLIGHT SEGMENT

One or more contiguous flight legs of an airplane scheduled with the intent of completion without maintenance.

FLIGHT TOUR

One or more flight segments for a given airplane at the end of which a layover will follow for scheduled maintenance.

GATE

The area where an airplane is parked to load and unload passengers and cargo.

HARDWARE

Physical devices, including computing, and memory media sensors and actuators.

HEURISTIC

A procedure for solving a well-defined mathematical problem by an intuitive approach in which the structure of the problem can be interpreted and exploited intelligently to obtain a reasonable solution.

INTEGRATION

The combination of separately implemented system functions with a view to simplifying or enhancing control laws and system hardware.

IRREVOCABLE ASSIGNMENT TIME

The time prior to departure after which no substitution of an airplane allocated to a flight can take place unless a preflight failure occurs.

k-OF-n SYSTEM

A system comprising n components or software algorithms, at least k of which must be operating in order for the system to perform its required functions.

LINE MAINTENANCE

Maintenance that is performed while the airplane is parked at the gate.

LINE REPLACEABLE UNIT (LRU)

A unit that is capable of being changed during line maintenance.

MINIMUM DISPATCH QUANTITY

That value of k, in a k-of-n stage, that must be met or exceeded to dispatch the subject airplane to a particular flight leg.

MOST DEMANDING FLEET DISPATCH QUANTITY

The highest value of k, in a k-of-n stage all over flight legs, that must be met or exceeded to dispatch the subject airline.

MOST DEMANDING STATION DISPATCH QUANTITY

The highest value of k, in a k-of-n stage, over all flight legs originating from a particular station that must be met or exceeded to dispatch the subject airplane.

NULLIFICATION

A nonfailed, nonoperating state or dependent effect arising from a failure elsewhere. There are two nullifications: unenergized (or cold), energized (or hot). In both cases, the component's functions are unavailable to the FCS.

ON-CONDITION MAINTENANCE (OC)

A preventive primary maintenance process that requires an appliance or part to be periodically inspected or checked against some appropriate physical standard to determine whether or not it can continue in service. The purpose of the standard is to remove the unit from service before failure during normal operation occurs.

OVERHAUL TIME LIMIT OR HARDTIME (HT)

A preventive primary maintenance process that requires an appliance or part to be periodically overhauled according to the carrier's maintenance manual or that it be removed from service. Time limitations may be adjusted based on operating experience or tests.

PARAMETER

Any of a set of properties, not under the designer's control, whose values influence the characteristics or behavior of a system.

PARENT STATION

A station that provides emergency maintenance for stations without maintenance facilities.

PAYBACK POINT

The amount of time taken to recover an investment as a result of income or savings derived from the investment.

PRESENT EQUIVALENT VALUE

The amount of money that could be invested now at a specified rate of return that would be equivalent to a fixed amount at a given time in the future.

RECONFIGURATION

Reassignment of tasks and components or both within a system to prevent internal faults from causing erroneous system output or to check the integrity of the system.

REDUNDANCY

The existence of more than one means for accomplishing a given function. Each means of accomplishing the function need not be identical.

REPLICATION

The means of performing a function with multiple, identical components or software.

RESIDUAL VALUE

The estimated worth of an asset or property at the end of its planned life.

RESTORATION

That work (on or off the airplane) required to return an item to a specific standard.

RESTORATION POOL

One or more airplanes under repair at a line station.

RETURN ON INVESTMENT (ROI)

A measure of the profitability of an enterprise based on the ratio of revenues minus expenses to investment. ROI may be computed in several different ways that should be stated when the ratio is used.

ROTABLE

An item that can be economically restored to a specified condition (usually the same as new) and in the normal course of operations is repeatedly rehabilitated over a period approximating the life of the flight equipment to which it is related.

SELECTION POOL

One or more airplanes that are not under repair and that are not irrevocably assigned to a flight leg.

SHIPPING CODE

A code used by the CEM to undertake one of 10 possible shipping actions detailed in Section 5.7.1.4.

SHOP REPLACEABLE UNIT (SRU)

A component that can be removed and replaced only under repair shop conditions.

SPECIAL POOL

One or more previously cancelled airplanes that have been restored to the most demanding fleet dispatch requirements.

SOFTWARE

(1) Computer programs, procedures, rules, and possibly associated documentation concerned with the operation of a data processing system. (2) The nonhardware part of computing that controls the use of the hardware.

STAGE

A set of identical components to which k-of-n logic applies.

STAGE MEMBERSHIP

The descriptive data that denotes the stage to which a component belongs at a given point in time.

STATUS CODE

A code used by the CEM to indicate which of six operating states listed in Section 5.7.2.2, item 3, apply to a particular component.

SUBJECT FCS OR AIRPLANE

The FCS or airplane under analysis in the CEM (sec. 5.2) as opposed to an FCS or airplane included in order to load shared resources in the CEM.

SUBSTITUTION

The act of substituting a more suitable airplane of the same model.

THROWAWAY OR EXPENDABLE

Components for which no authorized repair procedure exists and whose cost of repair would normally exceed that of replacement.

TIME MONITORING

See Overhaul Time Limit.

VALIDATION

The determination of correctness of the final program or software by testing against the design requirements in the user's environment.

VERIFICATION

The demonstration of consistency, completeness, and correctness of the software during the development life cycle.

VIRTUAL AIRPLANE

A physically nonexistent airplane assigned to a flight leg as a means of accounting for cancelled flights and their consequences during the simulation described in Section 5.0.

VIRTUAL FLIGHT

A cancelled flight leg served by a virtual airplane.

3.2 ACRONYMS

ACES	Airline Cost Estimating System
AOG	aircraft on ground
AOM	Analytical Optimization Method
ATE	automatic test equipment
BGU	bus guardian unit
CAB	Civil Aeronautics Board
CBOM	Cost and Benefit Optimization Model

CEM	Comprehensive Evaluation Model
CFA	cash flow analysis
CM	condition monitoring
CMC	critical minimum complement
CO, CO_2	cost of ownership
CPU	central processor unit
EROI	equivalent return on investment
FAA	Federal Aviation Administration
FCS	flight control system
FIFO	first-in-first-out
FTFCS	fault-tolerant flight control system
FTMP	Fault-Tolerant Multiprocessor
FTS	fault-tolerant system
GSE	ground support equipment
IEEE	Institute of Electrical and Electronic Engineers
KMS	kilogram, meter, second units
LCC	life-cycle cost
LRU	line replaceable unit
LSI	large-scale integration
MTBF	mean time between failure
MTE	manual test equipment
O&M	operation and maintenance
OC	on-condition maintenance
PEV	present equivalent value
PFTRT	programmable fault-tolerant remote terminal
ROI	return on investment
ROM	read-only memory
SIFT	Software Implemented Fault Tolerance
SRU	shop replaceable unit
TCO	total cost of ownership
TDS	triplex-duplex-simplex
TMR	triple modular redundancy

4.0 ANALYTICAL OPTIMIZATION METHOD

The method described in this section provides a means of rapidly and economically evaluating the many values of variables and parameters that are associated with the design and operation of commercial fault-tolerant flight control systems (FTFCS). The method is called the Analytical Optimization Method (AOM) and consists of taking an FTFCS concept that meets all functional and safety requirements and allocating additional resources to minimize costs. Figure 15 illustrates the allocation process. The arrows show the precedence relations in the mathematical model for design optimization.

There are essentially five activities to which the designer may allocate resources to reduce costs:

- Higher quality equipment (better reliability, packaging, fault isolation, durability, and less space or weight)
- Additional redundancy
- Additional spares provisions at line stations or depots
- More frequently scheduled maintenance (for restoration)
- More repair capacity (to reduce repair time)

These activities are represented as the decision variables for the economic optimization. Other activities for reducing life-cycle costs were not included in the model because of a lack of data or quantitative methods or because they were not the most important economic variables.

Depending on the particular airline environment, the level of resources allocated to each of these five activities may yield a significant reduction in cost for a given FTFCS. An assumption is made that all averages remain constant throughout the life of the system such as: failure rates, fixed fleet size, identical routes, interest rate, and passenger utilization. This assumption is necessary to have a common yardstick to compare various design and maintenance alternatives, but it also entails some error in absolute (as opposed to relative) accuracy of the cost estimate. Furthermore, it implicitly excludes non-steady-state management policies. An example of a non-steady-state policy is to start out with more spares than needed and allow inventory reduction or increase in fleet size to absorb the loss due to condemned spares.

The steady-state assumption provides the ability to derive a mathematical model that can be useful in preliminary design work for optimizing, screening, or ranking a wide range of aircraft systems under a wide range of environmental conditions. A steady-state model accomplishes this objective, however, at the possible expense of being able to accurately predict the actual costs and benefits of any specific system or of being able to evaluate non-steady-state maintenance policies that might use forecasted environmental changes or adaptive management control systems to further reduce costs.

ORIGINAL PAGE IS
OF POOR QUALITY

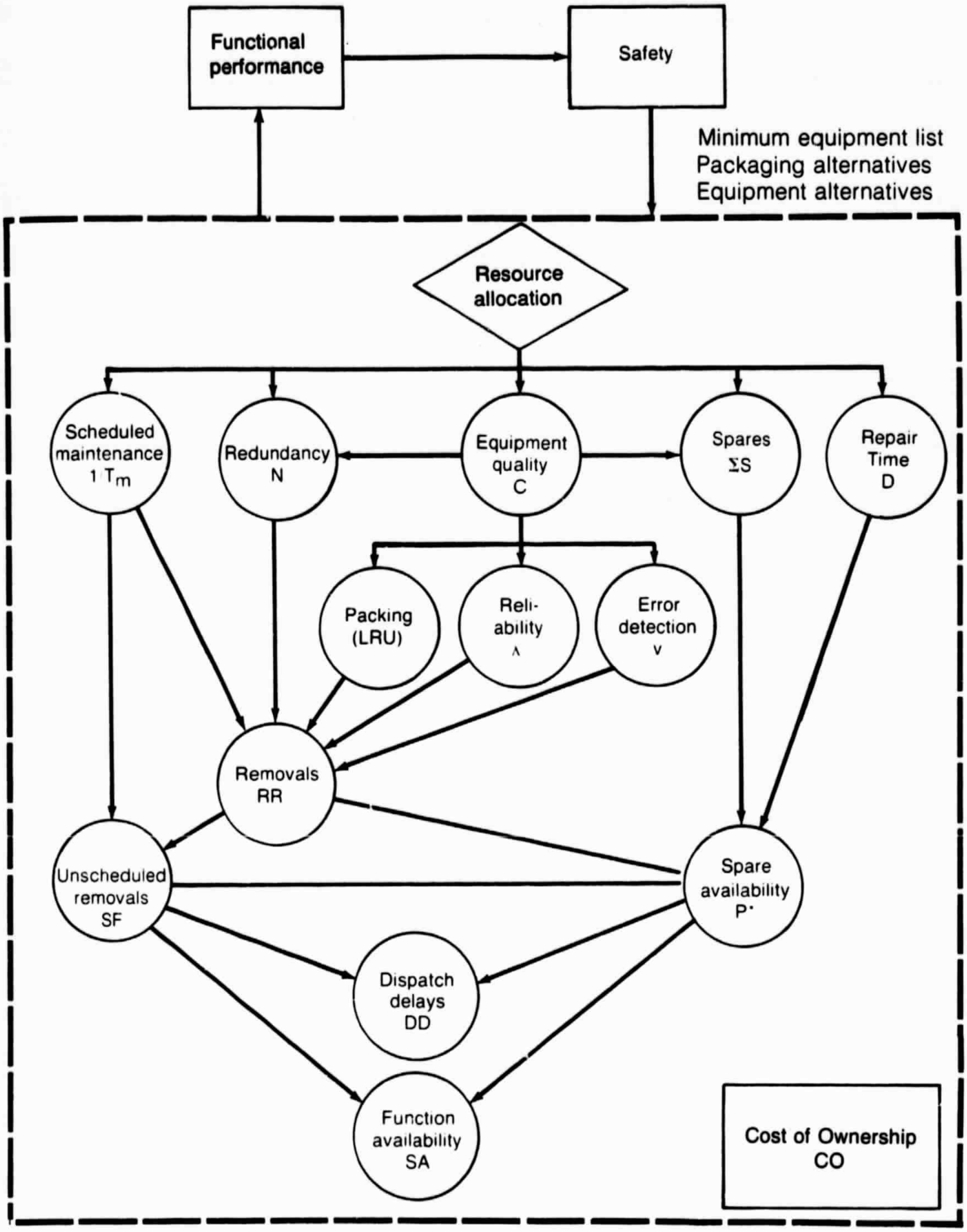


Figure 15. FTFCS Analytical Optimization Method

4.1 PRINCIPLES OF THE ANALYTICAL OPTIMIZATION METHOD

An FTFCS consists of a set of line replaceable units (LRU), many of which are redundant. Design optimization essentially looks at these LRUs and does the following:

- Determines optimum redundancy levels for each LRU (above that required for performance or safety)
- Determines optimum number of spares (both locations and quantities)
- Predicts average LRU demand (for removals, replacements, and repairs)
- Predicts average dispatch delay incidents

Analysis is performed at the lowest possible level for which failure rate data is available and for which independence holds. The lowest design level is assumed to be the component. An independent LRU can be optimized and evaluated independently. In general, each component will belong to two sets: a functional stage and an equipment LRU. These two sets correspond to the two major parts of the dispatch availability analysis. The first part of the analysis, described in Section 4.2, applies probability equations to functional stages. The second part of the analysis, described in Section 4.3, applies multiechelon inventory equations to the LRUs. These two parts of the analysis are then integrated by a cost minimization, described in Section 4.4.

4.1.1 Decomposition

The approach described of partitioning the FTFCS to a low level (considering data availability and dependencies created by packaging or series connection) and then performing the optimization at the lowest practical level is called decomposition. When optimization must be done simultaneously on two or more LRUs, the size of the state space increases exponentially, whereas if the optimization can be done on one LRU at a time, the size of the state space increases only linearly with the number of LRUs. Decomposition also provides more detailed information. For example, the contribution of each LRU to expected costs, maintenance workload, and dispatch delays is a useful byproduct of the analysis.

The appropriate decomposition of an FTFCS for analysis is done by the analyst. It requires judgment, taking into account the system architecture and function, and the quality of the available data. The typical problem will come to the analyst with a minimum equipment list that defines the components that make a system dispatchable. The natural starting point for decomposition is to define functional stages as the components in the minimum equipment list.

There are several ways in which LRUs can be dependent. An LRU is a fault containment region (i.e., a failure in the LRU by a short circuit would not normally create voltage overloads and failures in other LRUs). Two major kinds of dependency are accounted for in the present model:

- When two or more independent functions are contained in a single LRU
- When one LRU cannot function without the operation of another

4.1.2 Subsystems

There are four distinct kinds of subsystems that the model can optimize:

- The single-component LRU: the LRU that is not internally fault tolerant and is independent of other LRUs and fully reconfigurable in its stage.
- The multicomponent LRU: the LRU that contains two or more independent components in one package. This creates a physical dependency (even though the components are functionally independent). For example, because of packaging, the most demanding component will determine the redundancy of the other component in the same LRU.
- The multi-LRU component: the component that consists of two or more LRUs that are connected in series. In this case, the series of LRUs together is the component that is optimized, rather than the individual LRUs.
- The fault-tolerant LRU: the LRU that contains internal redundancy and the ability to function in the presence of failures. In this case, the LRU is modeled as a set of parallel (redundant) components. The mathematics for this case have an additional application for evaluating nonexponential components.

FTFCS optimization is defined in terms of the minimum expected cost of ownership, and all possible combinations of decision variables are evaluated and ranked in order to select the minimum overall cost for a given FTFCS configuration.

4.1.3 Stages

An FTFCS is described by a set of independent functional stages connected in series, each of which must be working for the FTFCS to work. Each functional stage consists of one or more identical components in a k-out-of-n parallel relationship and the working status of the stage may be defined by the number of working components it contains.

4.1.4 Components

The component is the basic building block of the analytical system. It is assumed that each component has known characteristics, including cost and failure rate. Component failure rates are assumed to be independent of time over the life of the system. Each component belongs to exactly two sets:

- A functional stage
- An equipment LRU

In the simplest case to analyze, the component is identical to the LRU, in which case the optimal configuration for the stage may be determined independently from all other stages in the system.

In fault-tolerant systems it is possible to have dependencies between components. For example, in the FTMP computer each LRU contains several different compo-

nents: power supply, clock, I/O port, memory, and processor. In these cases, the redundancy level and spares protection level cannot be set for each stage independently but must be determined by the minimum sum of costs over the feasible region of combinations (implying that the optimal design for a set of dependent components may have one or more stages that are not optimal).

4.1.5 Configuration Examples

Components are defined by the analyst in any way as long as they are independent and have a constant failure rate and belong to exactly one functional stage. A component may not be redundant in itself (because this would violate the constant failure rate assumption), but an LRU may contain redundant components.

The four types of subsystems are—

- A single component LRU that might belong to a stage that requires a minimum of six components for dispatch with each component packaged in its own LRU. In this case, the functional stage would be identical to the set of LRUs (a common case).
- A multicomponent LRU that might contain components from two functional stages, with a minimum of four components from one stage and six from the other stage required for dispatch. For this packaging arrangement, it is possible to have a surplus of components from one of the functional stages.
- A mutli-LRU component that might contain two types of LRUs, where three LRUs of each type are required for dispatch. In this case, the LRUs are in series, and the dispatch minimums must be the same for each type.
- A fault-tolerant LRU that might contain two redundant components in each LRU. In this case, even if three components are required for dispatch, the double packaging means that at least four must be installed on the aircraft.

4.2 DISPATCH DELAY ANALYSIS

This section presents the logic and mathematics used to compute dispatch delay incidents. Additional detail on the formula derivations is contained in Appendix B (ref. 5). These computations apply to each basic subsystem. There are slight variations required for each of the four types of subsystems (sec. 4.1.2).

For each LRU, let

m = minimum number of working components (LRUs) for dispatch, $m \geq k$, where k is the minimum number of components required for the stage to function

n = design level of the stage for maintainability, $n \geq m$ (not necessarily the optimum n)

r = the number of components that must fail in order for the stage to reach dispatch minimums ($r = n - m$)

RR = the expected number of LRU removals and replacements per year for the airline's fleet

T = the average number of operating hours between maintenance stops at which system is restored (not necessarily the optimum T)

λ = component failure rate = $\frac{1}{\text{MTBF}}$ failures/block hour (not necessarily the optimum λM)

F = annual number of operating hours of the airline's fleet (block to block)

ND = expected number of stage nondispatch incidents per year for the airline's fleet. (An incident is defined as the event that a stage drops below dispatch minimum.)

DD = expected number of dispatch incidents per year for entire fleet that result in delays or cancellations

P = the probability that a replacement LRU is available at an aircraft's next stop (not necessarily the optimum P)

The variables generally will have different values for different LRUs. Of the above variable, m, F, and T are assumed to be fixed parameters; n, λ, and P are assumed to be decision variables; and RR, ND, and DD are assumed to be system performance measures. The objective of the probability calculations is to calculate ND and DD as a function of the above parameters and decision variables. Dispatch delays are measured by the expected number of times in a year that a stage drops below its dispatch minimum at an airport that has no spare in stock, whether or not there is an actual time delay. A nondispatch condition at an airport without a spare can be a serious incident that requires extraordinary action, either by borrowing parts from another aircraft or by holding the aircraft on the ground until parts can be acquired. Not all such incidents result in a delay (for example, it may be the last flight of the day and can be repaired overnight), and not all delays are caused by such incidents (a delay may be due to lack of a trained mechanic even though a spare is available). In FTFCS optimization, however, the major decision variables probably depend on the level and location of line station spares; therefore, nondispatch incidents can be used in the optimization tradeoff as a reasonable proxy for actual dispatch delays and cancellations.

The following formula is used as a predictor of dispatch delays and cancellations:

$$DD = ND * (1-P) \tag{4-1}$$

DD is the expected number of times a year that the stage will be in a nondispatch condition at an airport where a spare LRU is not immediately available. The probability of two or more stages causing a single dispatch delay is sufficiently low that it does not need to be modeled.

Three maintenance policies are included in the model and may be selected by the analyst:

- Maintenance policy 1. The stage restoration to new condition takes place only at the main base at regular scheduled maintenance stops. The maintenance interval is supplied as input to the program in conjunction with the standard airline profiles. Separate algorithms also are available in Appendix E to find the optimum maintenance interval for a single stage).

- Maintenance policy 2. The stage restoration to new condition takes place on every overnight stop at airports that carry spares. The average interval between these restorations is computed internally by the program because it depends on the number of bases that carry spares (one of the optimization variables).

- Maintenance policy 3. The stage restoration to new condition takes place at the main base at regularly scheduled maintenance stops or at a line station whenever the airplane becomes nondispatchable.

Different maintenance policies (or options) may be assigned to different LRUs. It is assumed that at least emergency maintenance (to restore dispatch minimums) is done whenever a stage becomes nondispatchable.

It is assumed that at time zero the stage begins fully restored to a level n and must experience exactly (n-m) failures without intervening repair in order to reach the dispatch minimum. The time required to first reach dispatch minimum is a random variable Y, and the expected amount of time the stage must continue to function after it reaches the dispatch minimum is also a random variable $Z = T-Y$, where T is the expected interval between restorations. T is determined by the maintenance policy. By definition, the stage remains in dispatch minimum level m throughout the time Z because if there were another failure it would be repaired on an emergency basis (assuming maintenance policy 1). The stage cannot drop below dispatch minimum at any time except during a Z interval when it is operating at level m. The stage fails exponentially during these Z intervals, at rate $m^*\lambda$. The expected duration of the Z interval depends on the maintenance redundancy level (n-m) and on the restoration interval T.

Let A (T,n) = the expected number of block-to-block hours per restoration cycle during which the n-level stage will be operating at dispatch minimum conditions.

Then the expected number of stage failures (i.e., nondispatch incidents) is

$$ND = (m * \lambda) * \frac{F}{T} * A(T,n) \tag{4-2}$$

which is simply the rate of stage dispatch failure for stages operating at the minimum dispatch level times the total expected amount of time (per year) in which a stage would be at the m-level.

A(T,n) is computed by means of a the following formula:

$$A(T,n) = \int_{o}^{T} (T-t)\, f_y(t)dt$$

where $f_Y(t)$ is the density function of the distribution of Y, the time to first reach dispatch minimum status. Because the optimal maintainability redundancy will be significantly less than the reliability redundancy, i.e., $(n-m) \ll m$, the density function is closely approximated by that of the distribution of the sum of (n-m) independent, identically distributed, exponential, random variables. This is a gamma distribution that has a very simple analytical solution. Thus Y is to be represented as a random variable with a gamma distribution:

$$\text{mean } (\mu) = \frac{r}{\beta}$$

$$\text{variance } (\sigma^2) = \frac{r}{\beta^2}$$

$$\text{density } (f(t)) = \frac{\beta e^{-\beta t} (\beta t)^{r-1}}{(r-1)!}$$

where

$\beta = m * \lambda + \frac{(r+1) * \theta}{2}$ λ = average failure rate

$r = n-m$

θ = the degree to which redundancy is designed with hot spares, $0 \leq \theta \leq 1$ ($\theta = 1$ for all hot spares, the usual case).

Thus,

$$A(T,n) = \int_0^T (T-t) \frac{\beta e^{-\beta t} (\beta t)^{r-1}}{(r-1)!} dt,$$

that has the following analytical solution obtained by integration by parts (see App. B for derivation):

$$A(T,n) = T - Te^{-\beta T} \sum_{k=1}^{r} \frac{(\beta T)^{k-1}}{(k-1)!} + rTe^{-\beta T} \sum_{k=1}^{r} \frac{(\beta T)^{k-1}}{k!} - \frac{r}{\beta}(1-e^{-\beta T}) \tag{4-3}$$

Setting $\theta = 1$ represents spares that are carried hot, and $\theta = 0$ represents spares that are carried cold. The above calculations (equations 4-1, 4-2, and 4-3) would be repeated for each stage and each level of feasible redundancy.

To obtain the estimate of dispatch delays from equation 4-1, it is necessary to know P (the probability that a replacement LRU is available at an airplane's next stop). P, however, is a function of the level and distribution of spares provisions. This dependence raises an interesting characteristic of the optimization problem. There are two logical parts to the problem (a stage analysis to determine demand for emergency repair and expected dispatch delays and an LRU analysis to determine spares locations and inventory levels), but neither part can be done first. The following heuristic solution was adopted: if P is assigned an arbitrary value,

both parts of the analysis can be completed, and the analysis may be repeated for another assigned value of P. If this process is repeated for a range of values of P, the optimum can be derived by taking the minimum cost over the range of P, thereby giving the optimal value for P in the process. P is largely determined by the number of airports that stock spares, and the feasible values for P tend to be discrete with large jumps for $P < 0.50$.

For each stage (i) and each feasible redundancy level (n), equation 4-1 would be computed for each feasible P. It is estimated that approximately 20 to 30 values of P would need to be evaluated for a typical airline. For 3 feasible redundancy levels, 2 feasible maintenance policies, and 20 feasible values of P, the total number of probability evaluations for each stage would be 120 (= 3*2*20).

Maintenance policy 2 differs from the other two because the time between scheduled restorations is not fixed but instead is a random variable whose distribution depends on the distribution and level of spares. The estimation formula for dispatch delays (equations 4-2 and 4-3) assume that T is constant. In order to use the same equations for maintenance policy 2, it is necessary to calculate the expected restoration time and to assume that the variance in T is not large when compared to the mean.

For maintenance policy 2, the assumption is made that once every day the aircraft will have sufficient time in its schedule for restoration (e.g., overnight stops), and that restoration will be done if the stop happens to be at an airport that carries and has available the needed LRU spares. This assumption makes the expected restoration interval E(T) a function of P.

Let

L = average flight hours per airplane per day.

Then when an aircraft leaves an airport following restoration, the next restoration would occur on the following day (or L flight hours later) with probability P. It would occur on the second day (or 2*L flight hours later) with probability (1-P)*P, and so forth. Thus, the expected restoration interval E(T) (in flight hours) for maintenance policy 2 is:

$$E(T) = L*P + 2*L*(1-P)*P + 3*L*(1-P)^2*P + \ldots$$

This series converges and therefore can be computed as follows:

- Multiply the series equation by (1-P) and subtract:

$$E(T)-(1-P)*E(T) = E(T)*P = L*P + L*P*(1-P) + L*P*(1-P)^2 + \ldots$$

- Multiply this equation by (1-P) and subtract:

$$E(T)*P - (1-P)*E(T)*P = E(T)*P^2 = L*P$$

Thus, $E(T) = \frac{L}{P}$ (4-4)

This completes the description of the dispatch delay analysis. The mathematics

are powerful because—

- They provide precise probabilistic measures of the contribution of each stage to dispatch delay problems. This can not be done as well with either a simulation model or historical data because of the sample size requirements for small probabilities.

- They parameterize a number of important variables. Thus, hot/warm cold spares, various maintenance policies, redundancy levels, and restoration intervals all can be varied merely by changing equation parameters. This can not be done as well with a Markov reliability model because the state space would have to be changed as well as the parameters.

It is possible to include more equations to analyze different maintenance policies, components with nonexponential failure distributions, or systems that cannot be modeled by a series-parallel representation. As the complexity of the model increases, however, its computational efficiency and its reliability are likely to decrease.

4.3 SPARES PROVISIONING ANALYSIS

This section presents the mathematical model for the spares provisioning optimization of LRUs. Consumable parts are not repairable, and the mathematics of this section should not be applied to them. Different (and much simpler) equations are appropriate to determine inventory levels for consumable items (sec. 6.1.3).

The algorithms chosen for the equipment analysis, or the initial spares provisioning optimization of LRUs, are largely based on Reference 6. The problem is unique because it is multiechelon and because the number of locations is unspecified. The model described in this section works satisfactorily, but it may be possible to achieve greater computational efficiency.

An LRU is assumed to be a piece of equipment that can be quickly removed and replaced on an aircraft and is expensive enough to warrant repairing and returning to service. The LRU worth repairing generally requires tight inventory control, and for high-cost items, the optimal line station ordering policy is to order a replacement each time an LRU is used. The optimal ordering policy for LRUs (S,S-1) is commonly followed by airlines. LRUs are assumed to account for most of the FTFCS inventory investment.

The basic spares provisioning model is shown in Figure 16. In the center of Figure 16 is the depot where an inventory is maintained (perhaps at a zero level) and from which all shipments of restored LRUs are made to line stations. There are N line stations, J of which stock the LRU. Jay may be determined by the otpimization, but N (the potential number of line stations) is an input parameter, depending on the airline route. Each line station J with inventory has its own inventory level S_j (S_o = depot inventory). Each time an LRU is taken out of an aircraft, it is sent to the central repair shop and simultaneously a replacement LRU is ordered from the depot. Replacements come only from the depot, but emergency demands at stations without spares are met by another line station that has a spare.

ORIGINAL PAGE IS
OF POOR QUALITY

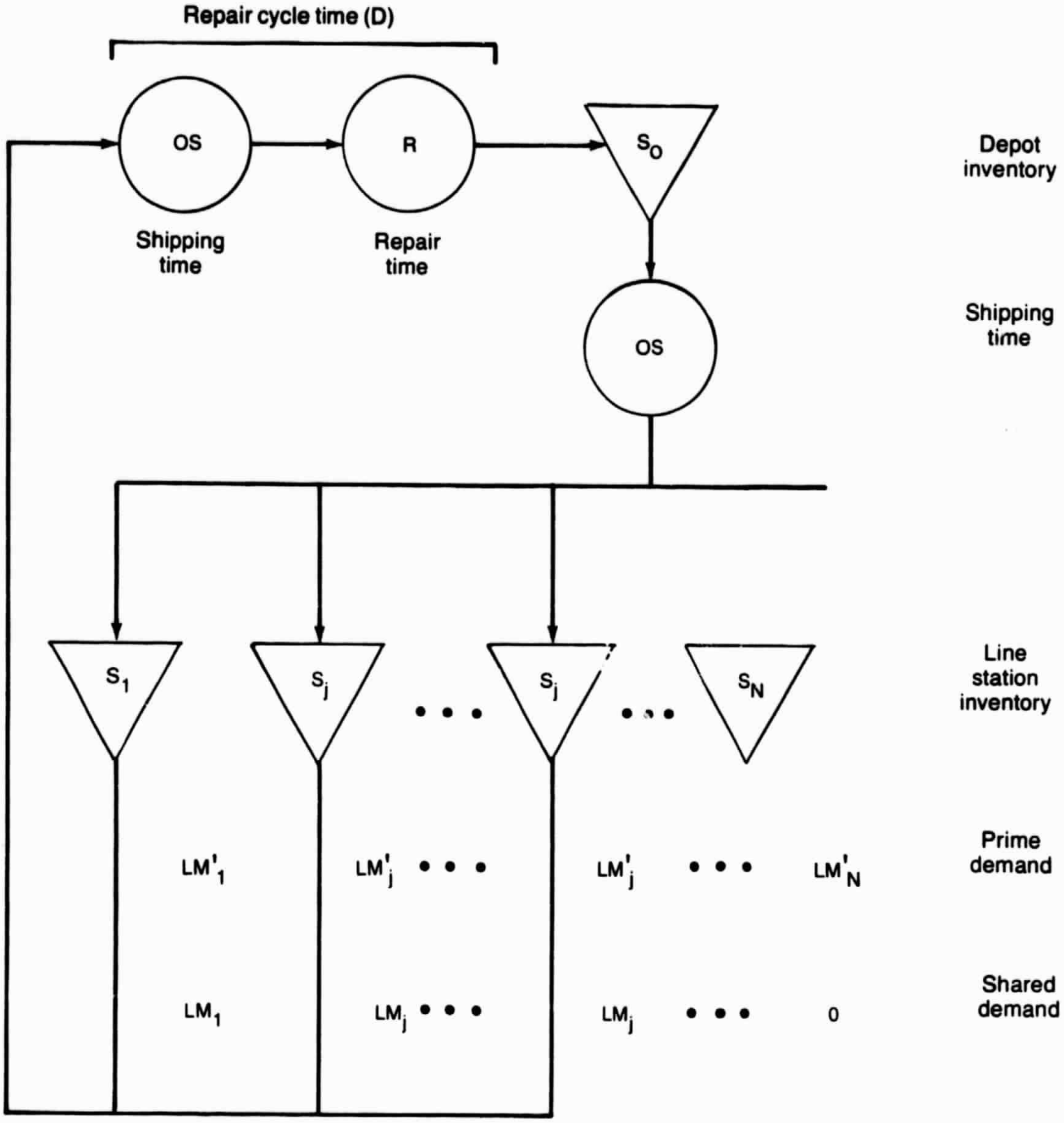


Figure 16. LRU Spares Provisioning Optimization

Line station 1 will represent the main base, and under maintenance policy 1, all scheduled maintenance is assumed to take place there. Thus, the demand on the main base (LM_1) will consist of all demand arising from scheduled maintenance plus a portion of the emergency demand. The emergency demand that occurs at bases without inventory is assumed to be distributed to the bases with inventory in proportion to demand.

As soon as an order is placed to the depot for a replacement, the depot ships an LRU if one is in stock. Otherwise, there is an additional wait if the LRU must be backordered from the depot. Order and shipping time (OS_i) is in calendar days, with a value for each airport. Typical time for OS is several days. Only the average order and shipping time influences optimal inventory levels, not the minimum time nor the distribution of time.

The total demand for parts (LM) is an output from the functional stage (dispatch delay) analysis of Section 4.2, depending on redundancy levels, flight hours, and maintenance policy. LM is the total average daily demand on the repair shop for replacement LRUs and is independent of how the LRU demand is distributed to the line stations.

The distribution to the line stations of the total LM is based on the assumption that, because of random failure rates, the LRU demand is equally likely to occur anywhere in the air. Thus, the demand at a given airport would be proportional to the percentage of all flight hours on flights that land at that airport. The airport loading factors are easily derived from the airline schedule, or they may be approximated (if flight hour data is not available) by the proportion of the total terminating block hours, the proportion of total departures, or the proportion of total arrivals at each airport.

Since the FTFCS model is a steady state model, if flight hours for a given airline are highly seasonal, then inventory should be based on the high season demand or there will be too many delays due to part shortages. Costs, on the other hand, depend on the annual repair workload, so the appropriate data for estimating life-cycle costs would be annual.

The repair cycle time (D) is the sum of the shipping time from the line station to the repair shop plus the time to be processed through the repair shop and placed in the depot inventory ready to be shipped to the line.

The following model uses a well-known principle of queuing network theory: for a Poisson demand process with immediate reordering and independent repair cycle times, both the state probabilities and the optimal inventory are independent of the distribution of the repair cycle time (ref. 7).

For each type of LRU, let

OS = number of days of administrative and shipping time from an airport to the repair shop or from the depot to an airport

R average days an LRU remains in repair shop

D = repair cycle time, $D = OS + R$

ORIGINAL PAGE IS OF POOR QUALITY

J = number of airports with LRU spares

S_j = number of spares stocked at airport (j)

LM = total LRU demand on the system per day

LM_j' = prime demand on airport (j), whether or not inventory is available

LM_j = shared demand on airport (j) for arriving flights and any other stations it supports

The total demand process is exponential because each individual LRU demand is exponential, and the random proportioning of the exponential demand preserves the exponential property. Thus, the demand process at each line station with inventory as well as the demand process at the repair shop are Poisson processes for which direct analytical solutions are possible.

Let

$T(S_o)$ = average resupply time for a line station, given a depot inventory of S_o, and

$W(S_o)$ = average number of days waiting for an ordered LRU to come into the depot inventory.

Then

$$T(S_o) = OS_i + W(S_o) \text{ and} \tag{4-5}$$

$$W(S_o) = (1/LM) * B(S_o) \tag{4-6}$$

where the average number of back orders is

$$B(S_o) = \sum_{i=S_o+1}^{\infty} (i-S_o) \frac{(LM*D)^i * e^{-LM*D}}{i!} \tag{4-7}$$

The derivation of equation 4-7, based on the probability that the LRU demand during the repair cycle time (D) is exactly i (because the component failures are assumed Poisson), is

$$P(\text{demand in time } D = i) = \frac{(LM*D)^i}{i!} e^{-LM*D}.$$

For a demand of $i \leq S_o$, there would be no backorder. For a demand of $i = S_o+1$ in time D, there would be one backorder; for a demand of $i = S_o+2$, there would be two backorders; and so forth. Therefore, the average number of backorders is

$$B(S_o) = \sum_{S_o+1}^{\infty} (i-S_o)*P(\text{demand in } D=i),$$

which is equal to $B(S_o)$ in equation 4-7.

For purposes of computation, equation 4-7 can be manipulated algebraically to an equivalent summation that involves finite terms and then combined with equaitons 4-6 and 4-5 to obtain the following:

$$T(S_o) = OS + \frac{1}{LM}*\left[LM*D - S_o - \sum_{i=1}^{S_o} (i - S_o)* \frac{(LM*D)^i*e^{-LM*D}}{i!} \right] \qquad (4\text{-}8)$$

For illustration, $T(S_o)$ was computed from equation 4-8, assuming a total demand rate (LM) of 1.04 LRUs per day and an average shipping time (OS) of four days.

The following shows how resupply time decreases as depot inventory increases or as repair cycle time decreases.

		REPAIR CYCLE TIME (D)				
		12	10	8	6	4
DEPOT INVENTORY (S_o)	0	16	14	12	10	8
	2	14	12	10	8	6
	4	12	10	8	6	5
	6	10	8	7	5	5
	8	8	7	5	5	4
	10	7	5	5	4	4

The entries of $T(S_o)$ have been rounded to integers. Note the difference between repair cycle time D and resupply time $T(S_o)$. These are two different concepts that have different effects on the system. The purpose of depot inventory is to reduce the line station's resupply time, and all line stations are affected by the presence or absence of a depot inventory.

Implicit in the repair shop is another inventory that is not shown in Figure 16; namely, the backlog inventory of failed LRUs waiting to be repaired. The purpose of the backlog inventory in the repair shop is to smooth the workload. Backlog may or may not be present. As far as the inventory optimization is concerned, the variable of interest about the repair shop is the average number of days that an LRU remains in the shop (including waiting time). The significant factor that one or more weekends may intervene should be accounted for in evaluating overall resupply times.

If equation 4-7 is evaluated for $S_o=0$; i.e., no depot inventory, then the expected number of backorders is

$$B(0) = \sum_{i=1}^{\infty} i* \frac{(LM*D)^i}{i!} *e^{-LM*D} = LM*D.$$

Thus, as expected, the average backorders in the repair cycle (D) would be the same as the average demand. All demand is backordered if there is no depot inventory. From equations 4-5 and 4-6, the corresponding resupply time in this case would be, as expected,

$$T(S_o) = OS + D$$

The fill rate for each line station is defined as the long run average proportion of LRU demands that can be filled imemdiately from stock. Because the demand process is assumed to be Poisson, and the resupply time $T(S_o)$ is computed for any given depot inventory S_o, the fill rate for line station j is a function of its inventory level.

In other words, for station j, the fill rate can be computed directly from the Poisson probabilities:

$$F(S_o,S_j) = \sum_{k=0}^{S_J-1} \frac{(LM_j*T(S_o))^k}{k!} *e^{-LM_j*T(S_o)} \tag{4-9}$$

If the fill rate for each line station (j) is weighted by its share of the demand (LM_j), then the average fill rate is determined. The average fill rate is, by definition, the spares coverage factor (P), the basic parameter that links the functional stage analysis and the LRU equipment analysis.

Thus, the average fill rate for any distribution of spares is:

$$F(S_1,S_2,\ldots S_j;S_o) = \frac{1}{LM}* \sum_{J=1}^{J} LM_j*F(S_o,S_j) \tag{4-10}$$

Note that the average fill rate calculated by equation 4-10 depends not only on the number of spares (S) but also on how they are distributed. Therefore, the fill rate that defines P should be the maximum average fill attainable from a given inventory level S; i.e., the maximum fill rate that can be obtained by the distribution of S spares. This maximization is a difficult problem that has yet to be fully solved. The following heuristic is incorporated in the basic model and a line of improvement is suggested in Appendix C. The heuristic is to constrain each line station to either have no LRUs in inventory or to have sufficient LRUs to ensure a minimum fill rate specified by the analyst. A suggested default value based on

airline practice for minimum fill rate is 0.97. This constraint makes the problem solvable by defining the decision variable in terms of the number of airports to carry spares.

First, observe that $T(S_o)$ is independent of line station inventories; therefore, it may be computed for a range of values depot inventory (S_o). The lower limit of $T(S_o)$ as S_o increases is OS; therfore, a default limit is set (the suggested initial value is 1.1 *OS), and $T(S_o)$ is tabulared for S_o = 0, 1, ... until

$$T(S_o)/OS \leq 1.1.$$

This should amount to no more than 20 or 30 values for S_o.

Next is the problem of finding the minimum number of line stations so that the average fill rate is greater than or equal to the given spares coverage P. Because of the constraint that each line station have a minimum fill rate of 0.97, the average fill rate from the line stations will be approximately 0.97 times the proportion of LM represented by the prime LRU demand in the J stations. In other words, the dominant factor is the number of line stations rather than the fill rate per station, as long as one requires high fill rates such as 0.97. The requirement of high fill rates is not unreasonable because airlines are accustomed to them for individual equipment and because human nature makes inventory locations with low fill rates unstable.

Using the above heuristic, the minimum number of line stations J that are to stock LRU spares can be calculated directly from the airport loading list. The third set of calculations is to determine for each potential depot inventory (S_o), the minimum number of spares required to provide a fill rate (or protection level) of 0.97. The calculations are repeated for each S_o, and the minimum of total inventory (S) is taken over all S_o. The formula used to find the minimum S_j for a given S_o, so that fill rate is greater than or equal to 0.97 (or analyst supplied value), is

$$\mathrm{Min}(S_j;\, S_o), \text{ so that } F(S_o,S_j) \geq 0.97, \tag{4-11}$$

where $F(S_o,S_j)$ is computed from equation 4-9.

4.4 OPTIMIZATION OF COST OWNERSHIP

This section describes the formulas for cost of ownership estimation and for optimizing over the range of decision variables. Only variable costs enter into the optimization. Costs remaining fixed over the range of decisions may be included or excluded in the input data, but they have no bearing on the optimization.

4.4.1 Cost of Ownership

Optimization is defined as the policy that minimizes the average variable costs. Costs are divided into two categories:

- Investment cost (IC)
- Operating cost (OC)

ORIGINAL PAGE IS
OF POOR QUALITY

Only the variable part of costs is used for optimization. For convenience, all of the investment costs are assumed to occur at the beginning of the period. These costs are then amortized over the system operational life. All costs are computed under steady-state conditions. The following formulas compute expected variable cost of ownership using the standard form of capital recovery factor to amortize investment costs:

$$\text{EVCO} = \text{Min}\left\{ \text{OC} + \frac{a(1+a)^q}{(1+a)^q - 1} * \text{IC} \right\}$$

where

IC =		$\sum_{\text{LRU}} C_1 \times N_i$	equipment costs
	+	$\sum_{\text{LRU}} C_2 \times S_i$	spares cost
	+	$\sum_{\text{LRU}} C_3$	other investment cost
OC =		$\sum_{\text{LRU}} C_4 \times N_i$	fuel burn cost
	+	$\sum_{\text{LRU}} C_5 \times RS_i$	condemned LRU cost
	+	$\sum_{\text{LRU}} C_6 \times S_i$	spares handling cost
	+	$\sum_{\text{LRU}} C_7 \times RR_i$	remove, replace, repair cost
	+	$\sum_{\text{Stage}} C_8 \times DD_j$	dispatch delay penalty
	+	C_9	Performance Benefits

And:

EVCO = expected variable cost of ownership (annual)

IC = initial investment cost

OC = operating cost (includes replacement costs)

a = airline's minimum attractive rate of return

q = life of system (years)

N_i = number of LRUs of type i in aircraft (fleet)

S_i = number of spares of type i in inventory

RS_i = expected number of condemned LRUs of type i (annual)

RR_i = expected number of removals of LRU i (annual)

DD_j = expected number of dispatch delays caused by stage j (annual)

C_1-C_{10} = cost parameters (analyst input)

Note that all of the costs are related to certain key system variables, the most important of which are:

- N_i (the redundancy)
- S_i (the spares)
- RR_i (the removal rate)
- DD_j (the dispatch delays)

4.4.2 Optimization of Cost of Ownership Over P

All of the cost variables will have been computed by the equations in Sections 4.2 and 4.3 for each set of decision variables by this point in the computer program. For each set of decision variables, however, the two parts of the model could not be applied unless the spares coverage parameter (P) was known. Neither part of the model could be applied until the other part was completed, and the impasse is resolved by repeating both calculations for several values of P. That is, at first there is not a single EVCO that can be associated with a given vector of decision variables (X), but rather a whole set of conditional EVCOs depending on P. The life-cycle cost associated with each combination of decision variables is determined by taking the minimum of the EVCOs over the set of possible coverage factors P. This is an optimization within an optimization, which is done by complete enumeration. In other words, for any given FTFCS configuration and airline environment, the estimate of life-cycle costs requires an optimization of P, and this process must be repeated for each FTFCS configuration to be evaluated.

The logic for this optimization within an optimization is that all of the computations can be done by the computer without analyst intervention. To do so, special provision should be made in the code to account for the differences in the four types of subsystems (single component LRU, multicomponent LRU, multi-LRU component, and fault-tolerant (LRU).

For the single component LRU, P is the same for the component in the stage analysis and for the LRU in the spares provisioning analysis. For the multicompo-

nent LRU, the physical packaging means that P is necessarily identical for each component in the LRU. When this identity is properly accounted for, it reduces the number of internal iterations that are needed. The fault-tolerant LRU is handled in the same way as a single component LRU (the special characteristics of the fault-tolerant LRU are accounted for in the spares provisioning equations.)

The multi-LRU component has the special characteristic that in the stage analysis there is only one P, but in the spares provisioning analysis there are many alternative distributions of LRU spares by which the overall P may be obtained. This becomes, in effect, a "flyaway-kit" problem to determine the optimum distribution of spare LRUs to achieve a given P. The critical question concerning the solution is whether or not some airports will stock some but not all of the LRUs in the component. Experience suggests that this will be the case only if the LRUs in the component had major differences in cost or reliability. An exact solution to this problem is not currently available but is one of the possible model enhancements. However, there is satisfactory approximate solution. It seems clear that a multi-LRU component that had a low-cost, low-reliability LRU in series with a high-cost, high-reliability LRU is a bad design and thus unlikely to be a feasible design alternative. Therefore, it is reasonable to assume that the same P will be applied to each LRU in the multi-LRU component. This assumption does not force identical spares locations or identical spares levels, and it provides a reasonable approximation of the optimal solution for the multi-LRU component.

4.5 DEMONSTRATION OF THE PROCESS

This section describes the application of the mathematical model to analyze an FTFCS design. A theoretical FTMP computer design is used because component dependencies within the LRU cause it to be the most difficult to analyze. Hypothetical data is used for illustrating the process of analysis. A complete analysis is not practical until a computer code is available though the requirements for code and a host computer are modest in size (app. D).

The FTFCS design is assumed to be packaged in 10 identical LRUs, each containing:

- LRU power supply
- Clock
- I/O port
- Memory
- Processor

Let

k = critical minimum complement (the minimum number of components required for the system to function)

m = dispatch minimum complement (the minimum number of components required for aircraft dispatch)

MTBF = the component mean time between failure

Assume that the following data have been determined from the engineering or safety analysis:

COMPONENT	MTBF (hr)	k	m
Power supply	7 000	(Not applicable)	
Clock	100 000	5	8
I/O port	20 000	1	4
Memory	5 000	2	5
Processor	5 000	5	8

Furthermore, assume that these components have the following dependencies:

- If a power supply fails, all other components in the LRU become inoperable.
- If a processor fails, no other components are affected.
- If a memory fails, the processor in the same LRU becomes inoperable.
- If an I/O port fails, the memory (and thus the processor) in the same LRU becomes inoperable.
- If a clock fails, the I/O port (and thus the memory and the processor) in the same LRU becomes inoperable.

This represents a complex system because of the partial dependency of components within each LRU. The mathematical model applies to independent stages. Because none of the stages in this problem are independent, it is necessary to map the system into an equivalent, approximate series-parallel model. To validate this approach, data have been computed by a detailed Markov model that explicitly includes the dependencies. Comparison shows that the basic mathematical model can provide an extremely close estimate of the probability measures, even when applied to nonindependent components.

To use the mathematical model, the four independent stages and corresponding failure rates are defined as follows:

STAGE	FAILURE RATE
Clock	1/6542
I/O	1/4930
Memory	1/2482
Processor	1/1659

These failure rates were obtained by adding together all of the failure rates of components that can make the given component inoperable. For example, the memory is disabled by a memory failure, an I/O failure, a clock failure, or a power supply failure (1/2482 = 1/5000 + 1/20 000 + 1/100 000 + 1/7000).

Assume that the maintenance cycle time between stage restorations is 200 hours. Then the time (A) per cycle during which the stage is operating at the minimum dispatch level and the corresponding expected number of stage dispatch failures is

computed from formulas 4-2 and 4-3 in Section 4.2 as follows:

STAGE	m-LEVEL HOURS A (200,10)	DISPATCH INCIDENTS (ND)
Clock	2.44	0.0030
I/O	0.00	0.0000
Memory	0.02	0.0000
Processor	25.48	0.1229
	Total	0.1259

This shows that almost every time (98%) the FTFCS drops below dispatch minimums, the processor will be involved. It also shows that the total number of expected dispatch failures is 0.1259 per 200 flight hours, given a system redundancy level of 10 LRUs.

For a check, this same problem was analyzed as a discrete Markov process with 40 states, each state corresponding to the number of working components. The result of the discrete Markov analysis was:

P (single dispatch failure) = 0.0735
P (two dispatch failures) = 0.0194
P (three dispatch failures) = 0.0039
P (four or more failures) = 0.0007

When the above probabilities are weighted by the number of failures and totaled, the expected number of dispatch failures per 200 flight-hour cycle is 0.1268 (= 0.0735 + 2*0.0194 + 3*0.0039 + 4*0.0007). Thus, the delay rate calculated by equations 4-3 and 4-2 is quite close to the rate calculated by the Markov state space model (0.1259 compared with 0.1268, or 99.3% accuracy).

The dispatch delay analysis is repeated for other levels of redundancy with the following dispatch incidents per 200 hours:

	DISPATCH INCIDENTS (ND) PER 200 FLIGHT HOURS			
STAGE	n = 8	n = 9	n = 10	n = 11
Clock	0.2446	0.0308	0.0030	0.0002
I/O	0.0000	0.0000	0.0000	0.0000
Memory	0.0022	0.0003	0.0000	0.0000
Processor	0.9644	0.3759	0.1229	0.0352

In order to estimate the number of dispatch delays that would result from these dispatch incidents, it is necessary to specify the airline environment parameters. For this example, the airline data was based on a Pan American fleet of 43 Boeing 747 airplanes flying a route of 50 airports with the average load per airport defined in terms of the proportions of total departures from each airport.

Given the airline flight hours of 190 000 per year, the expected number of dispatch incidents for each redundancy level and the expected LRU removal and replacement workload can be calculated. The result is:

REDUNDANCY LEVEL (n)	NONEMERGENCY REPLACEMENT	EMERGENCY REPLACEMENT	TOTAL
8	0	285	285
9	195	111	306
10	296	36	332
11	343	10	353

As the redundancy level increases above dispatch minimums, the emergency removals steadily decrease (for a given maintenance interval), and the nonemergency removals steadily increase. The penalty associated with each emergency replacement depends on the proportion occurring at line stations that don't have spares. This is related to the spares coverage factor P.

Assuming that the average dispatch delay penalty is \$5000 per occurrence and that the related cost is \$1000 per replacement, then the following costs are computed for two values of P and four values of redundancy n:

P	n	REPLACEMENT COST (RR) \$000's	DISPATCH DELAYS \$ 000's	TOTAL COST RR + DD \$000's
0.5	8	285	713	998
0.5	9	306	278	584
0.5	10	332	90	422
0.5	11	353	25	378
0.8	8	285	285	570
0.8	9	306	111	417
0.8	10	332	36	368
0.8	11	353	10	363

For each P, the minimum number of line stations that must stock spares is determined, and the total spares level calculated. For example, for the given airline environment, 6 bases must be stocked at a fill rate of 0.97 in order to provide a spares coverage P = 0.50, and 17 must be stocked to provide a spares coverage P = 0.80. For a redundancy level of 10 LRUs, the emergency demand from the above table of 36 per year would be assigned to the 6 (of 17) bases in proportion to their traffic loading factor. The nonemergency demand of 296 per year would be assigned to the main base. For example, if the emergency demand at one base is 9.2 per year, then the inventory equation 4-9 can be used to calculate an inventory level of 3 LRUs (for a resupply time of 14 days). For the main base, an inventory of 19 LRUs would be required. If the 5 additional bases require a total of 6 more LRUs, then the total spares required for a spares coverage P = 0.50 is 28 LRUs. This calculation would be repeated for various depot inventory levels until the minimum spares are found for each spares coverage level P. The final optimization occurs by finding the minimum total life-cycle costs over P, using the formulas of Section 4.4. There are thousands of calculations

involved in this process, and it is not practical to do the search without a computer program.

The analysis of SIFT, FTMP, or any other FTFCS design proceeds along these same lines. SIFT uses a single-component LRU and thus, it is easily modeled for maintainability optimization. FTMP uses a multicomponent LRU similar to that described in this section. It can be modeled by the same approach used here.

In summary, it has been shown how the analysis proceeds by modeling the FTFCS as a set of independent stages, calculating the effective failure rates that apply to the components, calculating the probabilistic measures for each level of redundancy at or above dispatch minimums, calculating inventory requirements for selected spares coverage levels, and optimizing by an exhaustive search over all design alternatives to find the design with the minimum life-cycle cost. An extreme case of dependent components within an LRU was used to demonstrate the validity of the mathematical model for optimizing complex FTFCS designs.

4.6 INPUT AND OUTPUT FOR THE ANALYTICAL OPTIMIZATION METHOD

4.6.1 Input Data Requirements

The input data requirements are minimal, and most of the necessary data may be incorporated into the program as default values, capable of being overridden by the analyst. This use of default values allows the analyst to focus on design issues and to adequately account for maintainability without getting into those matters that are not related to design alternatives.

Optimal design may depend on the environment (i.e., large fleets or few airports tend to call for lower redundancy and more spares, whereas small fleets or many airports tend to call for higher redundancy and less spares). The airline environment data and the economic parameters can be stored in the computer program as default values and provided to the analyst as output values, together with sensitivity estimates derived by the model. The following input variables would be treated as default values:

- List of average traffic load for each airport on the route (fraction of average annual flight hours represented by the flights that are scheduled to land at a given airport)
- Number of aircraft type in the fleet
- Average annual flight hours and number of departures
- Dispatch delay penalty (estimate of the economic penalty of delays and cancellations)
- Fuel burn penalty due to weight or space
- Average order and shipping time between repair shop and each line station (for each LRU)

- Inventory management fee (costs other than capital cost for maintaining stores that is expressed as a percentage of spares investment)

- Minimum acceptable rate of return

- Life (the number of years of expected service life, over which all capital investments are amortized for the optimization)

Sensitivity analysis identifies any parameter for which a significant change in a default value has an impact on the optimal design, allowing the analyst to determine where better input data may be needed.

The normal input data that the analyst must provide is:

- FTFCS component functional and physical relationships
- MTBF (for each component)
- Dispatch minimum complement (for each component)
- Removal verification rate (for each LRU)
- Weight (for each LRU)
- Condemnation rate (for each LRU)
- Acquisition cost (for each LRU)
- Average cost of removal, replacement, and repair (for each LRU)
- Average time in the repair shop (for each LRU)
- Average interval between scheduled restorations (in flight hours)

4.6.2 Output

Once the the input data is entered, the optimization and cost analysis is accomplished without analyst intervention. The output from the analysis for the whole system and for each subsystem consists of:

- Optimal LRU spares levels (for each LRU)

- Optimal LRU spares locations (for each LRU)

- Optimal LRU redundancy levels (for each LRU)

- Expected number of removals, replacements, repairs per aircraft-year (for each LRU)

- Expected number of annual dispatch delay incidents (for each stage and for whole system)

- Investment cost (for each LRU and for whole system)

    - Equipment
    - Spares

- Operating cost (for each LRU and the whole system)

    - Fuel burn penalty (due to weight or space requirements)
    - Condemned spares replacements (for LRU and whole system)

- Remove, replace, and repair costs (for each LRU and whole system)
- Dispatch delay penalties (for each stage and whole system)
- Performance benefits (if supplied as input)

- Cost of ownership (investment cost combined with operation and maintenance (O&M) costs weighted in an appropriate manner for each subsystem and for whole system)

4.7 SENSITIVITY ANALYSIS

The AOM analyst is provided with sensitivity data for each of the variables or parameters, obtained by rerunning the model with one change at a time. Sensitivity is defined as the percentage change in optimal cost that results from a given change in a variable or parameter. This is a valuable screening and design tool because it helps to focus attention on which of the following entities whose variations have the most effect on the system:

- Component MTBF (for each component)
- Removal verification rate (for each LRU)
- Cold spare versus hot spare (for each LRU)
- Cost of capital
- Dispatch delay penalty
- Order and shipping time
- Average time in repair shop
- Interval between scheduled maintenance

Each sensitivity analysis requires rerunning all of the mathematics, thus the preceding list increases the computational requirements by approximately an order of magnitude when it is used.

In addition to these capabilities, the program will provide the analyst with sensitivity data that reflects the importance of the airline parameters on the optimal design variables. Rather than varying one airline parameter at a time, the best approach is to base the sensitivity index on two or three default airline profiles.

The analyst also may be interested in evaluating packaging alternatives to reflect various packaging arrangements. Packaging is largely a work of art that is determined as much by performance (engineering) constraints as by economics. General guidelines for packaging will evolve such as:

- Avoid packaging expensive, reliable components with inexpensive, unreliable components

- Avoid packaging expensive components together if they have significantly different minimum dispatch requirements

- Try to reduce the variety of LRUs when packaging two or more components together

5.0 COMPREHENSIVE EVALUATION MODEL

The Comprehensive Evaluation Model (CEM) is a plan for a Monte Carlo simulation designed to simulate the behavior of fault-tolerant flight control systems (FTFCS) from sensor to actuator. However, only events of possible economic value are simulated. The CEM can accommodate non-fault-tolerant as well as fault-tolerant systems (FTS), and designs can be represented hierarchically at system, LRU, and SRU levels. The simulated airline in which the FTFCS operates provides an environment in which queues can develop for resources such as spares, mechanics, and test equipment and can represent actual or hypothetical airlines. As airplanes satisfy a schedule of flights, flights can be delayed or cancelled. Such events and details of recovery from them are tallied, as are statistics on the resources used.

The analyst using the CEM will be able to simulate the real world at whatever level of detail chosen. The FTFCS can have components with constant or nonconstant failure rates and have hot or cold standby components. Dependencies among components can cause failure or nullification. Contention for airline resources required by other systems or other airplanes can be included or excluded, in the latter case, by letting the other systems have zero failure rates. Of course, the more factors that can be excluded from a given study, the easier analysis of the results becomes. Other things that may be seen in the description of the CEM are the following:

- Components that exhibit behavior analogous to wearout are time monitored so that they may be removed and replaced either at the overhaul time limit or on-condition, whichever comes first.
- The model can handle one-person and two-person maintenance tasks at line stations.
- One level of nesting is accommodated. An LRU may be contained within another LRU.
- The model handles throwaway parts and vendor-repairable parts.
- The downstream consequences of delays and cancellations are generated by the model.
- Line stations are permitted to make best airplane substitutions of one subject airplane for another.
- Aircraft on the ground (AOG) at nonmaintenance stations draw labor and parts from parent stations.
- The model exploits the unique characteristics of fault-tolerant redundancy management so that minimum dispatch rules, at the user's option, become a function of the flight leg.
- The model generates time delays for 10 different shipping situations.
- Delays and cancellations may propagate through the airline.

PRECEDING PAGE BLANK NOT FILMED

- The model user may specify expedited shipping times for emergency resupply.
- The model restores the FCS to the zero-failure condition at specified intervals.
- Priorities are assigned and changed as circumstances warrant at line stations and in the repair shop.
- Failures occurring in flight are permitted to become visible on a probabilistic basis either pre- or postflight.
- Line maintenance and repair shop task times are drawn from user-supplied probability distributions.
- Line maintenance other than removal and replacement is accommodated by introducing pseudo-LRUs that have zero weight and cost and infinite supply, with a remove and replace time distribution descriptive of inspection and adjustment times.
- The model enables different task times to be generated for removal of first and additional items. The user may specify less time to remove and replace additional identical items than that needed for the first item.

Other features of the model relate to scheduled maintenance, flight cancellations, and airplane flight allocation. Scheduled maintenance, because of its planned nature, has been treated simply and nonstochastically in the CEM. Finding a realistic method of simulating flight cancellations resulted in the development of the "virtual airplane."

A virtual airplane is created whenever a flight is cancelled. The virtual airplane substitutes for an AOG until the AOG is returned to service, at which time the virtual airplane is destroyed. The penalties of cancelled flights can thus be comprehensively accounted for.

The description of the CEM follows the modular construction of the CEM design and is illustrated by Figure 17. Several conventions have been adopted, and the ones for flow charts are shown in Figure 18.

Each block on a flow chart has a written description of its input, output, and processing method, in standard English, with the following conventions:

IF-THEN	Conditionally defines algorithms to be executed
IF-THEN-ELSE	Conditionally defines either of two sets of algorithms to be executed
UNTIL	Defines algorithms to be repeated until a condition is true
WHILE	Defines algorithms to be repeated while a condition is true
ENDIF	Terminates IF-THEN or IF-THEN-ELSE

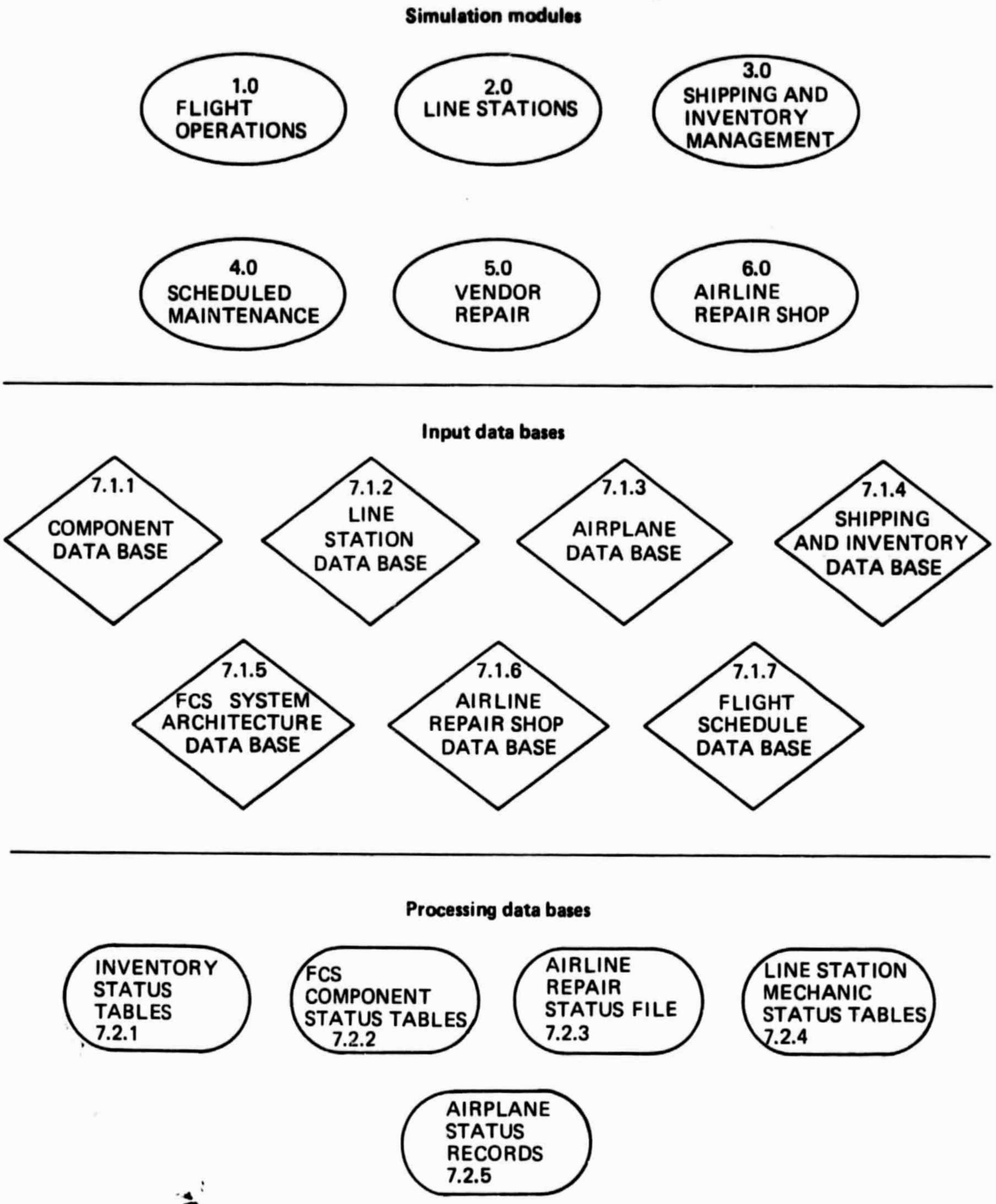


Figure 17. Major CEM Elements

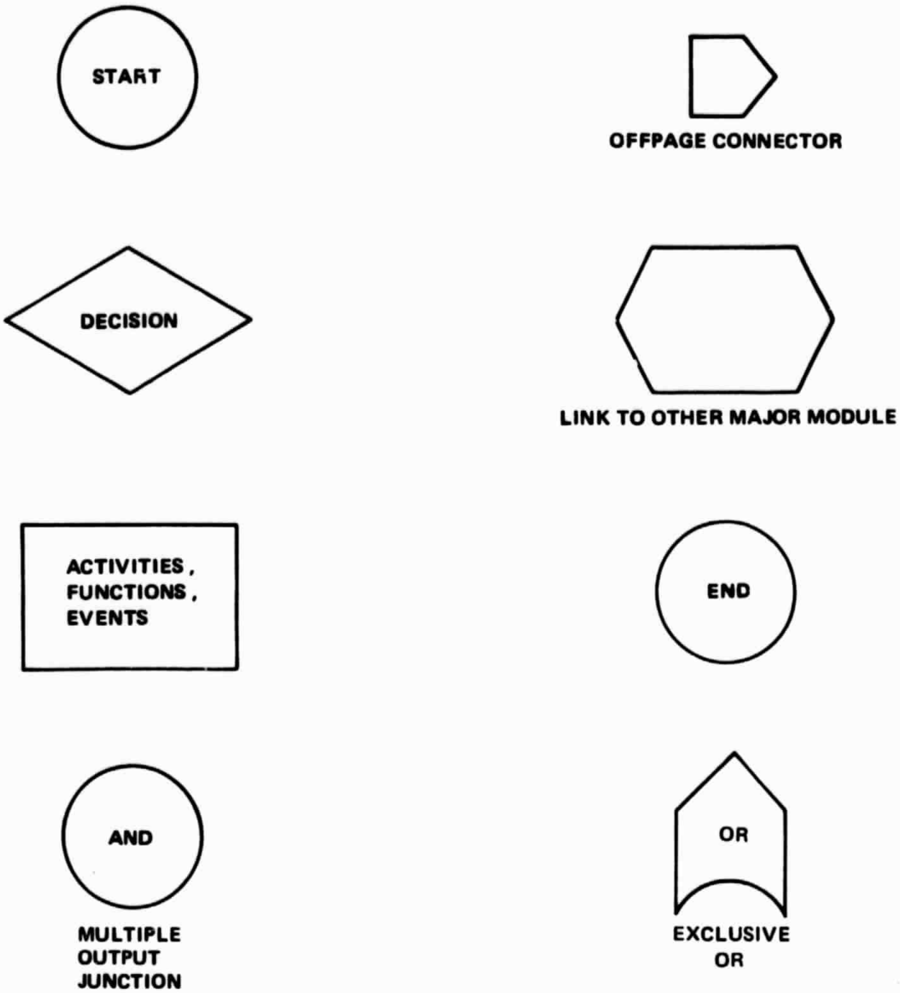


Figure 18. Flow Chart Conventions

ENDWHILE	Defines the end of a WHILE loop
ENDUNTIL	Defines the end of an UNTIL loop

Inequality symbols used throughout Section 5.0 are:

LE = less than or equal to
GE = greater than or equal to
NE = not equal to
GT = greater than
LT = less than

Each flow chart block that is not further expanded into a lower level flow chart is explained in subsections that specify its inputs, outputs, and processing requirements.

Input

A list of all input quantities together with their sources (a flow chart block, an input data base, or a processing data base) is shown here.

Output

Similarly, a list is provided of all output quantities and their destinations (a flow chart block or a processing data base).

Processing

The processing subsection specifies two basic functions performing numerical calculations or logical testing to control the simulation flow; some blocks accomplish both. Computational algorithms are displayed in pseudocode format, thus avoiding syntax peculiar to particular programming languages.

A number of pseudocode statements such as those that refer to the SIMULATION CLOCK, FLAGS, or STATUS RECORDS have been included to assist in understanding the simulation logic. High-level simulation languages possess simulation utilities that account for event timing, queue management, and record keeping and that simplify the task of programming.

Frequently, additional commentary and explanations are included where appropriate. It is required that each event or decision involving an airplane be time tagged to enable proper sequencing of parallel and merging activity streams.

NOTE: Both flow charts and input, output, processing descriptions contain numerous cross-references within Section 5.0. For brevity the leading "5" has been dropped, except for the main headings.

5.1 FLIGHT OPERATIONS SIMULATION

Primary Function. The primary function of the flight operations simulation is to generate failures in operating components of interest. Failures, dependent failures, and nullifications are generated here. Comprehensive pre- and postflight failure-state statistics are gathered to enable accurate assessment of operating benefits and penalties.

Simulating Continuously Operating Equipment. Some airplane equipment such as communications equipment operates nearly continuously, being turned on at the first preflight check of the day and turned off up to 16 hours later at the final arrival. The analyst will handle this by multiplying the nominal failure rate for continuously operating components by the ratio of total operating time to total block time for the subject airplane fleet. This approach will enable failures to be generated only during block time, thus simplifying the simulation logic.

Failure Visibility. Continuously operating equipment may fail in the interval between arrival and departure, with the failure becoming visible at the preflight check. Other equipment may also experience failure that becomes visible at this time. For whatever the reason, the model permits existing failures to mainifest themselves probabilistically at either the pre- or postflight check. Once visible,

failures remain visible until repaired.

Virtual Airplanes. Virtual airplanes are discussed at the beginning of the Line Stations Simulation, Section 5.2. Because they serve cancelled flights, virtual airplanes are not subject to failure.

The flight operations simulation logic, shown in Figure 19, follows:

1.1 Virtual Airplane?

Input

Virtual airplane flag: passed from 2.20

Output

YES: directs flow to 1.8
NO: to 1.2

Processing

IF VIRTUAL FLAG SET
THEN YES
ELSE NO
ENDIF

1.2 Subject Airplane?

Input

Subjet airplane flag: passed from 2.20

Output

YES: directs flow to 1.4
NO: to 1.3

Processing

IF SUBJECT AIRPLANE FLAG SET
THEN YES
ELSE NO
ENDIF

1.3 Generate Dispatch Critical Failures (for Other Airplanes)

This block generates dispatch critical failures in all the airplanes, other than the subject airplane, that comprise the airline's fleet.

Input

- Departure event notice: from 2.20, which includes:

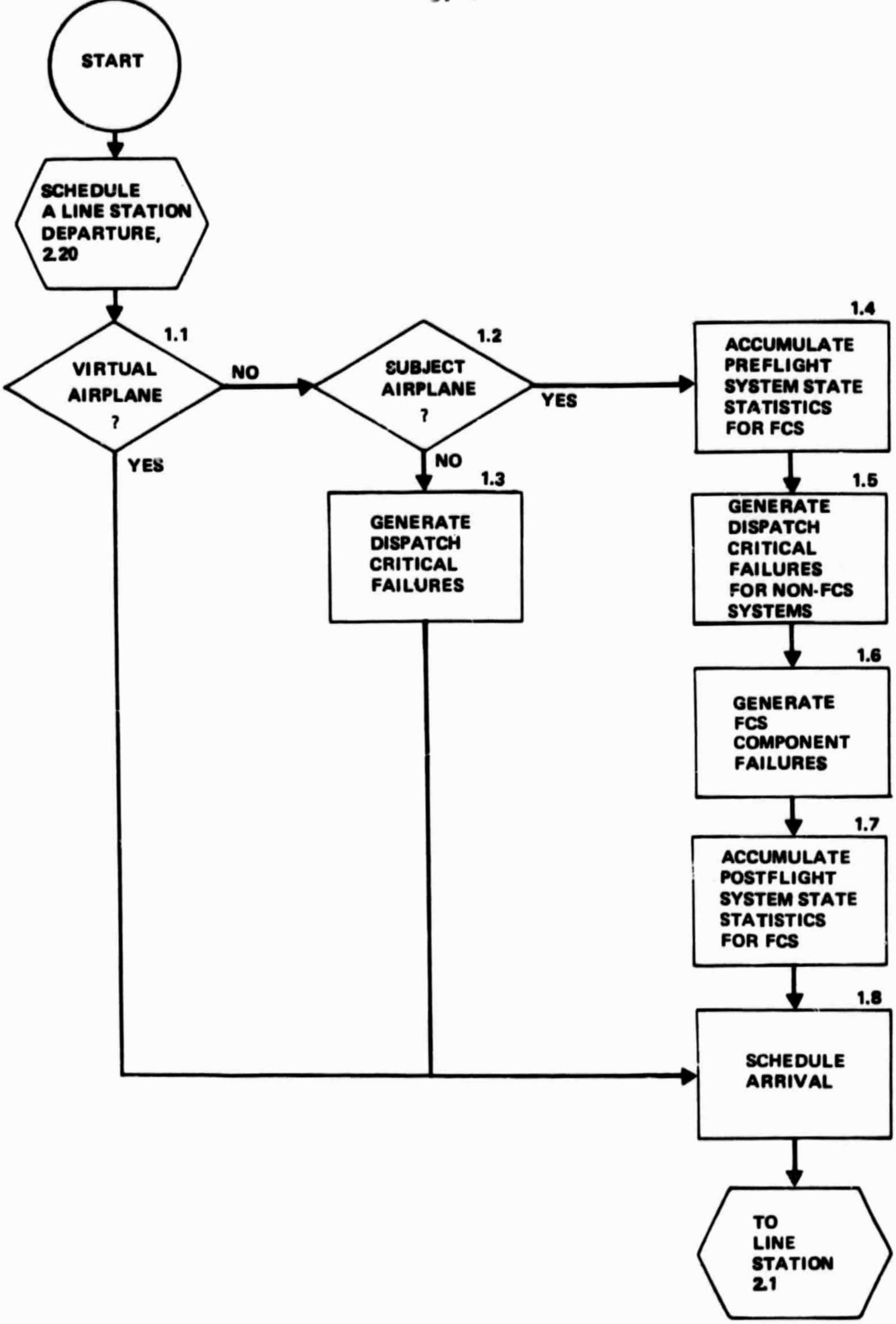


Figure 19. Flight Operations (Section 1.0)

- Tail number
- Airplane model
- Flight number
- Departure time
- Subject airplane flag (unset, in this case)

- Unscheduled removals per block hour from airplane data base, item 5.2

- Block time for flight leg: from flight schedule data base, item 5

Output

Quantity of failures during block time: to 2.8.2.4

Processing

- USING THE UNSCHEDULED REMOVAL RATE AND THE BLOCK TIME, COMPUTE THE EXPECTED NUMBER OF FAILURES AS

  EXPECTED NUMBER OF FAILURES =
  UNSCHEDULED LINE REMOVAL RATE PER BLOCK HOUR * BLOCK TIME

- USING THE EXPECTED NUMBER OF FAILURES AS THE MEAN OF A POISSON DISTRIBUTION, DRAW TO DETERMINE THE ACTUAL NUMBER OF FAILURES

1.4 Accumulate Preflight System State Statistics for FCS

Input

- Component status table (sec. 7.2.2)
- Stage membership file: from system architecture data base, item 2
- Affiliation file: from system architecture data base, item 1
- Preflight system state tallies for all FCS stages (sec. 7.2.2)

Output

- Update preflight system state tallies: used in flight performance analysis after completion of CEM simulation runs (sec. 6.2 in the main body of this document)

Processing

INCREMENT THE SUBJECT AIRPLANE FLEET TALLIES FOR K = 0, K = 1, ...K = N FOR EACH K-OF-N STAGE in the FCS

1.5 Generate Dispatch Critical Failures for Non-FCS Systems (on Subject Airplane)

Input

- Departure event notice: from 2.20, which includes

- Tail number
- Airplane model
- Flight number
- Departure time
- Subject airplane flag (set in this case)

- Unscheduled removal rate per block hour for non-FCS systems from component data base, item 5.1

Output

Quantity of non-FCS failures on subject airplane: to 2.8.2.3

Processing

- USING THE UNSCHEDULED REMOVAL RATE AND THE BLOCK TIME, COMPUTE THE EXPECTED NUMBER OF OTHER SYSTEM FAILURES AS
  EXPECTED NUMBER OF FAILURES =
  UNSCHEDULED LINE REMOVAL RATE PER BLOCK HOUR * BLOCK TIME

- USING THE EXPECTED NUMBER OF FAILURES AS THE MEAN OF A POISSON DISTRIBUTION, DRAW TO DETERMINE THE ACTUAL NUMBER OF FAILURES

1.6 Generate FCS Failures

The generation of failures for fault-tolerant systems in the CEM reflects the inherent peculiarities of these systems in an airline environment.

- Some failures cause other dependent failures and nullifications.
- Cold standby components may be activated if available.
- Not all failures are immediately visible.

This block is expanded in Figure 20 and the associated input-output-processing requirements follow.

1.6.1 Any Addresses Remaining?

A discussion of addressing is contained in Section 2.8.2.2; briefly, addressing is a means of referencing the physical location of an LRU or SRU within the FCS.

Input

Next unscanned address in component status table

Output

YES: directs flow to 1.6.2
NO: to 1.7

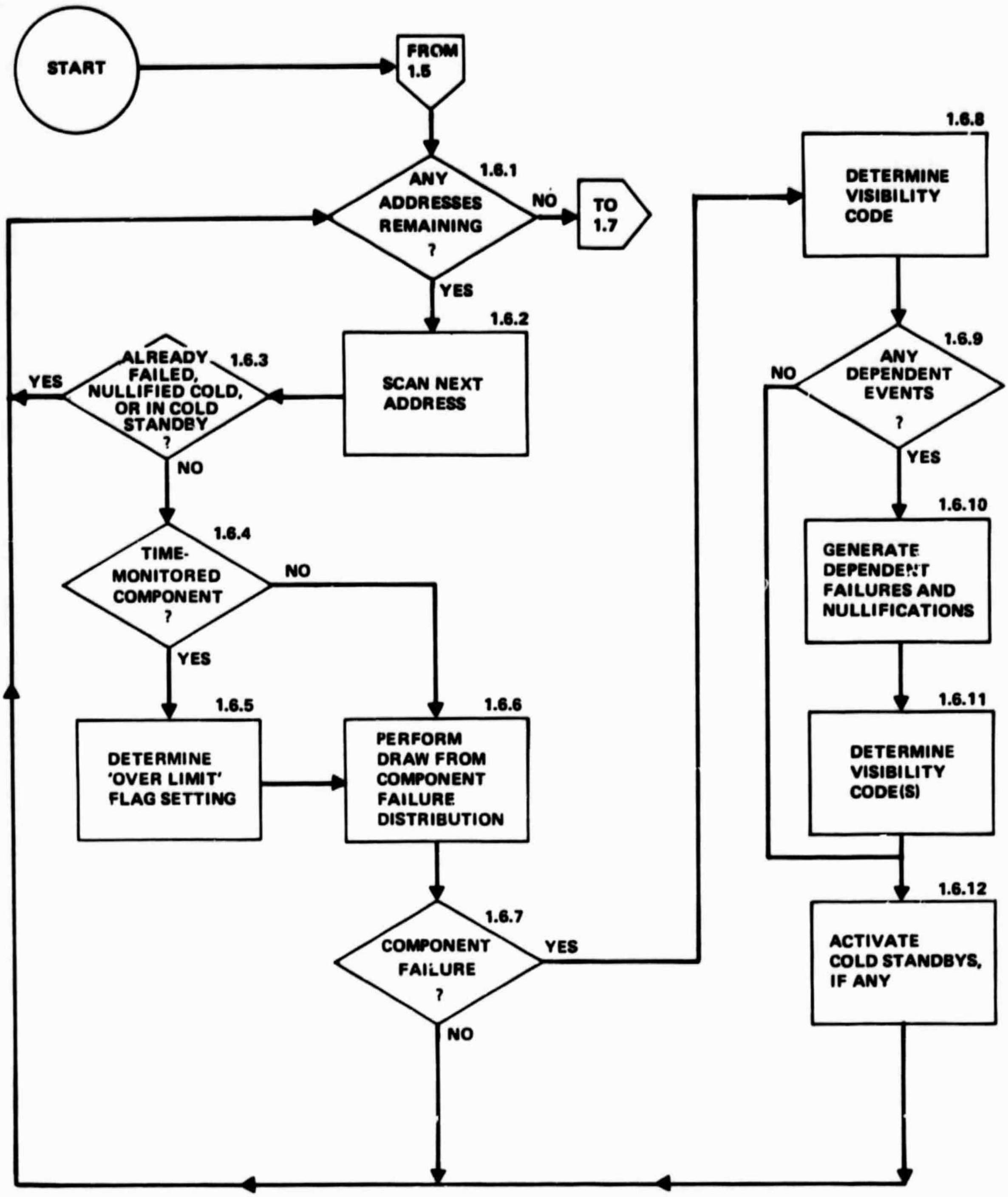


Figure 20. Generate FCS Component Failures (Section 1.6)

Processing

- IF NEXT ADDRESS = END-OF-FILE
  THEN NO
  ELSE YES
  ENDIF

1.6.2 Scan Next Address

Input

Next address in component status table

Output

New file pointer location: used internally

Processing

MOVE FILE POINTER TO NEXT RECORD

1.6.3 Already Failed, Nullified "Cold," or in Cold Standby?

Input

Status field: from current component status record, item 3

Output

Yes: directs flow to 1.6.1
No: to 1.6.4

Processing

- IF STATUS CODE = 2 OR 5 OR 6
  THEN YES
  ELSE NO
  ENDIF

1.6.4 Time-Monitored Component?

Input

Time-monitor flag in current component status record

Output

Yes: directs flow to 1.6.5
No: to 1.6.6

Processing

IF TIME MONITOR FLAG SET
THEN YES
ELSE NO
ENDIF

1.6.5 Determine "Over Limit" Flag Setting

Input

Operating time limit: from component status record, 5.2
Accumulated operating time: from component status record, item 5.3

Output

"Over limit" flag setting: to component status record, item 5.4

Processing

- IF ACCUMULATED OPERATING TIME GT OPERATING TIME LIMIT
  THEN SET "OVER LIMIT" FLAG
  ELSE CONTINUE
  ENDIF

1.6.6 Perform Draw From Component's Failure Distribution to Set Failure Flag

Input

- Component part number and address: items 2 and 1.1 in 7.2.2, component status table

- Accumulated operating time for time-monitored components: from component status record, item 5.3

- Failure rate versus operating time table for time-monitored components: from component data base, item 8.3; otherwise constant failure rate, item 8.4

- Departure event notice: fron [illegible]

- Nominal block time: from flight schedule data base, item 5

Output

Failure status code, item 3 in 7.2.2, component status table: to 1.6.7

Processing

- IF COMPONENT IS TIME MONITORED, LOOK UP FAILURE RATE, LAMBDA, AS FUNCTION OF OPERATING TIME: ELSE USE CONSTANT FAILURE RATE

P(SUCCESS) = EXP (-LAMBDA* BLOCK TIME)

ENDIF

- MAKE DRAW FROM UNIFORM DISTRIBUTION (0,1)

- IF DRAW VALUE GE

  P(SUCCESS)
  THEN SET STATUS CODE = 2
  ELSE CONTINUE

  ENDIF

1.6.7 Component Fail?

Input

- Component status code, item 3 in 7.2.2, component status table: updated in 1.6.6

Output

- Yes: directs flow to 1.6.8

- No: to 1.6.1

Processing

IF COMPONENT STATUS CODE = 2
THEN YES
ELSE NO
ENDIF

1.6.8 Determine Visibility Code

Input

- Postflight visibility proportion: from component data base, item 7

Output

- Visibility code: to 2.5 and 2.10

Processing

- DRAW FROM UNIFORM DISTRIBUTION (0,1)

- IF VALUE LE PROPORTION OF FAILURES VISIBLE AT POSTFLIGHT CHECKOUT
  THEN SET VISIBILITY CODE TO "POST"
  ELSE SET VISIBILITY CODE TO "PRE"

ENDIF

1.6.9 Any Dependent Events?

Input

Components status record, item 1.2

Output

YES: directs flow to 1.6.10
NO: to 1.6.12

Processing

- IF DEPENDENT EVENT FLAG SET
  THEN YES
  ELSE NO
  ENDIF

1.6.10 Generate Dependent Failures and Nullifications

Input

- Address of the failed component: item 1.1, component status table

- Linked list of affected addresses and event codes: from dependency effects table in system architecture data base, item 3

Output

Updated component status record for each affected address: to 1.6.11

Processing

UPDATE COMPONENT STATUS RECORDS PER DEPENDENT EFFECTS RECORD AS FOLLOWS:

- IF EVENT CODE = F, SET STATUS CODE = 2
- IF EVENT CODE = A, SET STATUS CODE = 4 (sec. 7.1.5, item 3)
- IF EVENT CODE = D, SET STATUS CODE = 5

1.6.11 Determine Visibility Codes

Input

- Updated component status records for components that have dependent failures: from 1.6.10

- Postflight visibility proportion: derived from component data base, item 7

Output

Visibility codes for failed components

Processing

SEE 1.6.8

1.6.12 Activate Cold Standbys, if Any

Input

- Failure addresses: from 1.6.6 and 1.6.10
- Stage membership file and dependent effects table: from FCS system architecture data base, items 2 and 3
- Component status records for components at addresses of interest

Output

- Exchange of addresses between failed components and their cold standby replacements

Processing

NOTE: When cold standby components are activated, they exchange addresses with the component they replace. Since cold standbys are invulnerable to failure or nullification (as modeled here), the existence of a failure code for a component located at a cold standby address indicates that the exchange has already been made and that a replacement, if any, must be found elsewhere.

```
WHILE FAILURE ADDRESSES REMAIN
      TEST THE ADDRESS AFFILIATION FILE FOR STAGE MEMBERSHIP
      IF NO STAGE MEMBERSHIP
      THEN EXIT
      ELSE WHILE OTHER STAGE ADDRESSES REMAIN
          IF COMPONENT STATUS CODE = 6
          THEN EXCHANGE COLD STANDBY AND
             FAILED COMPONENT ADDRESSES
          ELSE CONTINUE
          ENDIF
          ENDWHILE
      ENDIF
ENDWHILE
```

1.7 Accumulate Postflight System State Statistics for FCS

Input

- Component status table

- Address affiliation file: from system architecture data base, item 1

Output

- Updated postflight system state statistics: used in fuel burn performance analysis after completion of CEM simulation runs

Processing

INCREMENT THE SUBJECT AIRPLANE FLEET TALLIES FOR K = 0, K = 1, ... K = N FOR EACH K-OF-N STAGE IN THE FCS

1.8 Schedule Arrival

Input

- Departure event notice: from 2.20
- Block time: from flight schedule data base for flight leg of interest, item 5
- Local time zone correction: from flight schedule data base, item 2.2
- Accumulated block time by tail number: from airplane status record, item 9
- Accumulated operating time on time-monitored parts: from component status table, item 5.3

Output

- Arrival event notice with parameters as follows:
    - Airplane model
    - Tail number
    - Arrival time
    - Destination code
    - Through-flight flag setting
    - Accumulated block time on airplane at arrival time: used to update airplane status record (see 7.2.5)
    - Incremented operating time for time monitored components

Processing

- LOCAL ARRIVAL TIME = DEPARTURE TIME + FLIGHT DURATION + LOCAL TIME ZONE CORRECTION
- INCREMENT ACCUMULATED AIRPLANE BLOCK TIME FOR TAIL NUMBER OF INTEREST BY BLOCK TIME OF DEPARTING FLIGHT
- IF A THROUGH FLIGHT
  THEN SET THROUGH FLIGHT FLAG
  ENDIF

5.2 LINE STATIONS SIMULATION

- Each airplane arrival will trigger creation of an airplane status record that will be updated as the airplane moves through the simulation. The fields are specified in 7.2.5.

- Airplanes remain in the line station selection pool until irrevocably assigned to a flight leg.

- Airplanes awaiting restoration or being restored are in a "restoration" pool until all tasks are completed, at which time they enter the selection pool; however, an airplane in the restoration pool may be irrevocably assigned to a flight segment prior to completion of its maintenance tasks.

- To simulate the downstream effects of cancellations and the efforts undertaken to limit them, the CEM invokes the concept of virtual flying. A virtual airplane is created whenever a cancellation occurs. The virtual flight arrives on time at the next station. If the best airplane algorithm (2.6) cannot find another airplane in the selection pool or in the restoration pool, the virtual airplane will be dispatched again. If the best airplane algorithm does find a substitute airplane, the virtual airplane will enter the selection pool and will be employed only as a last resort. In this fashion, the CEM models the airline's utilization of locally excess capacity to preclude cancellations.

  Furthermore, this scheme ensures that each station's arrival and departure flow remains in balance.

  The CEM will tally all virtual flights, and the total number of virtual flights will in fact be the number of cancelled flights. The real airplane whose flight was cancelled will be assigned low maintenance priority prior to a user specified cutoff time, say 6 p.m., after which it will be assigned high priority and the airplane will be restored to the most demanding fleet requirement.

  After completion of restoration, the cancelled airplane remains unavailable until passage of an average deadhead or relocation time, after which it may be "instantly" substituted for the next arriving virtual airplane at any station.

  The replaced virtual airplane is then removed from the fleet ("destroyed"), thus terminating the cascade of cancellations.

  The logic for the line station flow, shown in Figure 21, follows:

2.1 Scheduled Maintenance Due?

Input

- Virtual flag setting: from airplane status record, item 4

- Subject airplane flag setting: from airplane status record, item 2

- Airplane block time accumulation by tail number: from airplane status record, item 9, updated in 1.8

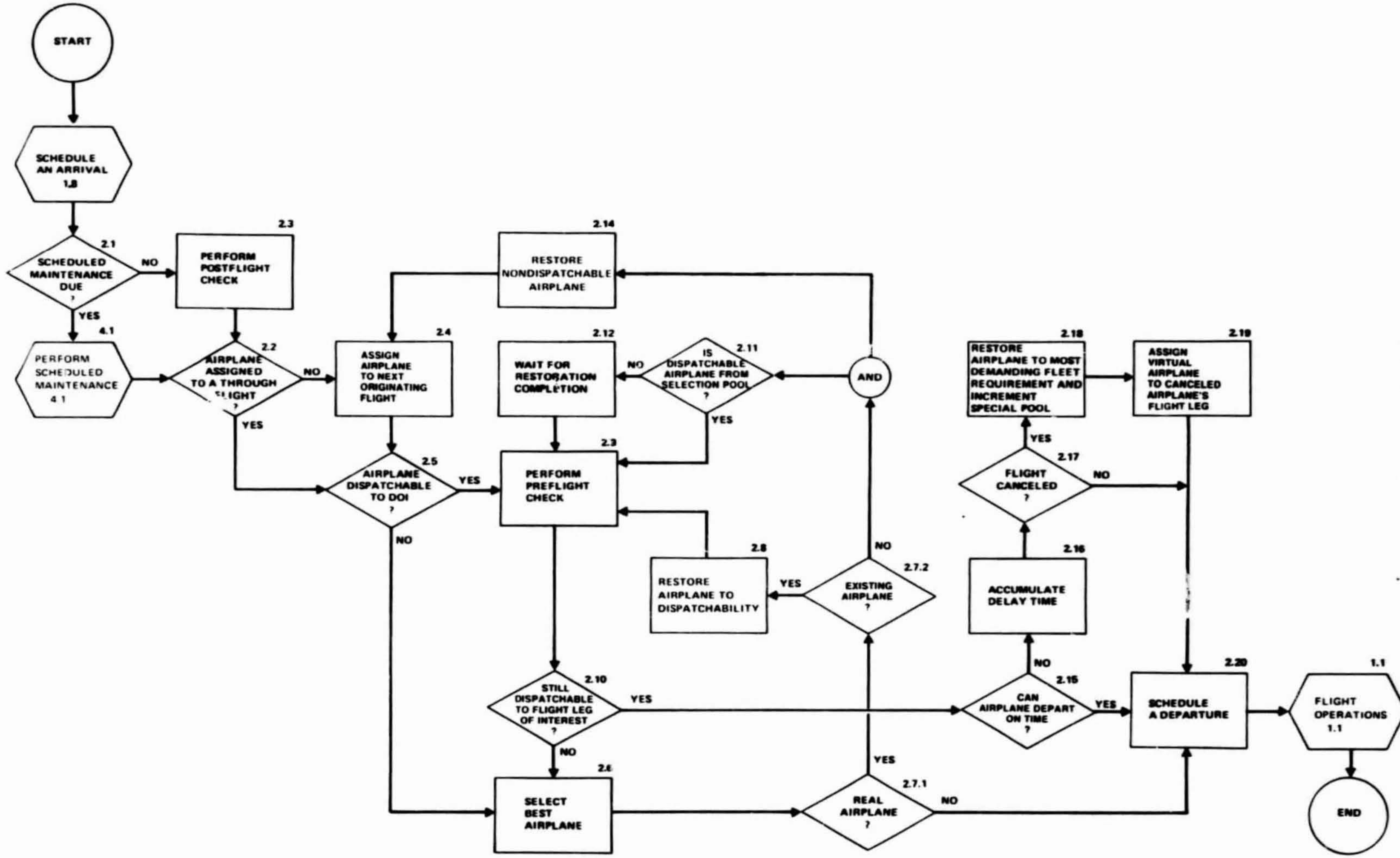


Figure 21. Line Station (Section 2.0)

- Scheduled maintenance interval for subject FCS: from airplane data base; item 2.2

Output

- YES: directs flow to 4.1
- NO: directs flow to 2.3

Processing

- IF VIRTUAL FLAG SET OR IF SUBJECT AIRPLANE FLAG UNSET

- THEN NO

- ELSE
    IF AIRPLANE ELAPSED BLOCK TIME IS GE
    FCS SCHEDULED MAINTENANCE INTERVAL
    THEN YES
    ELSE NO
    ENDIF
  ENDIF

2.2 Airplane Assigned to a Through Flight?

Input

- Through-flight flag setting: from 1.8

Output

- YES: directs flow to 2.5
- NO: to 2.4
- Association of airplane tail number to flight number: to 2.7.2 and 2.13

Processing

- IF THROUGH-FLIGHT FLAG SET
  THEN YES
  ELSE NO
  ENDIF

- ASSOCIATE TAIL NUMBER WITH DEPARTING FLIGHT LEG NUMBER AND ENTER IN AIRPLANE STATUS RECORD, ITEM 3.1

2.3 Perform Postflight Check

Input

- Postflight checkout time: from airplane data base, item 4
- Arrival time: from airplane status record, item 3.7, updated in 1.8
- Airplane model: from airplane status record, item 1.2
- Line station type: from line station data base, item 13.1

- Virtual airplane flag setting: from airplane status record, item 2

Output

- Checkout completion time: used internally to enable start of maintenance
- Incremented total of checkout delay time by airplane type and over all types
- Incremented number of arrivals by airplane type
- Clock advance to postflight check completion time: to 2.8.5 and 2.8.26

Processing

- DEFINE CHECKOUT WAITING TIME AS TIME INTERVAL BETWEEN ARRIVAL TIME AND BEGINNING OF CHECKOUT

- IF VIRTUAL FLAG SET, PERFORM CHECK IN ZERO TIME WITH ZERO LABOR AND ZERO WAITING TIME
  ENDIF

- INCREMENT NUMBER OF ARRIVALS OVERALL

- INCREMENT TOTAL POSTFLIGHT CHECKOUT WAITING TIME FOR THE STATION, AT LINE MAINTENANCE STATIONS ONLY

- ADVANCE CLOCK TO COMPLETION OF POSTFLIGHT CHECK

2.4 Assign Airplane to Next Originating Flight

Input

- Airplane model: from airplane status record, item 1.1
- Airplane tail number: from airplane status record, item 1.2
- Time now: internally generated
- Flight schedule data base: items 1, 2.1, 3, 4, and 6

Output

- Association of airplane tail number to flight number: used in 2.13
- Departure time: to 2.20

Processing

- THROUGH FLIGHTS ARE PREASSIGNED

- WITH DEPARTURES IN CHRONOLOGICAL ORDER, ASSIGN AIRPLANE TO NEXT UNASSIGNED ORIGINATING FLIGHT NUMBER FOR AN AIRPLANE OF ITS MODEL DESIGNATION

2.5 Is Airplane Dispatchable?

Input

- Airplane model: from airplane status record, item 1.1

- Airplane tail number: from airplane status record, item 1.2
- Updated component status table: from 1.4
- Minimum dispatch quantities by stage for flight leg: default values from system architecture data base, item 2.3; override values from line station data base, item 5
- Address affiliation file: from system architecture data base, item 1
- Availability criteria file: from system architecture data base, item 4
- Virtual flag setting: from 2.19

Output

- YES: directs flow to 2.9
- NO: to 2.6

Processing

```
IF VIRTUAL FLAG SET,
THEN NO
ELSE WHILE STAGES REMAIN
     EXTRACT QUANTITY UNFAILED FROM COMPONENT STATUS
     TABLE
     IF VISIBILITY CODE = "PRE," IGNORE THE FAILURE
     ELSE COMPARE WITH MINIMUM DISPATCH QUANTITY AS
     FOLLOWS:
         IF QUANTITY UNFAILED GE MINIMUM DISPATCH QUANTITY,
         THEN YES
         ELSE NO
     ENDWHILE
ENDIF
```

2.6 Select Best Airplane

This block is expanded in Figure 22.

It is intended that this block be invoked whenever an airplane is undispatchable or if the airplane is virtual.

2.6.1 Augment Local Selection Pool

Input

- Tail number: from airplane status record, item 1.1
- Airplane model: from airplane status record, item 1.2
- Flight number to which assigned: from airplane status record, item 3.1
- Virtual flag setting: from airplane status record, item 2
- Local selection pool list for airplane model of interest

ORIGINAL PAGE IS
OF POOR QUALITY

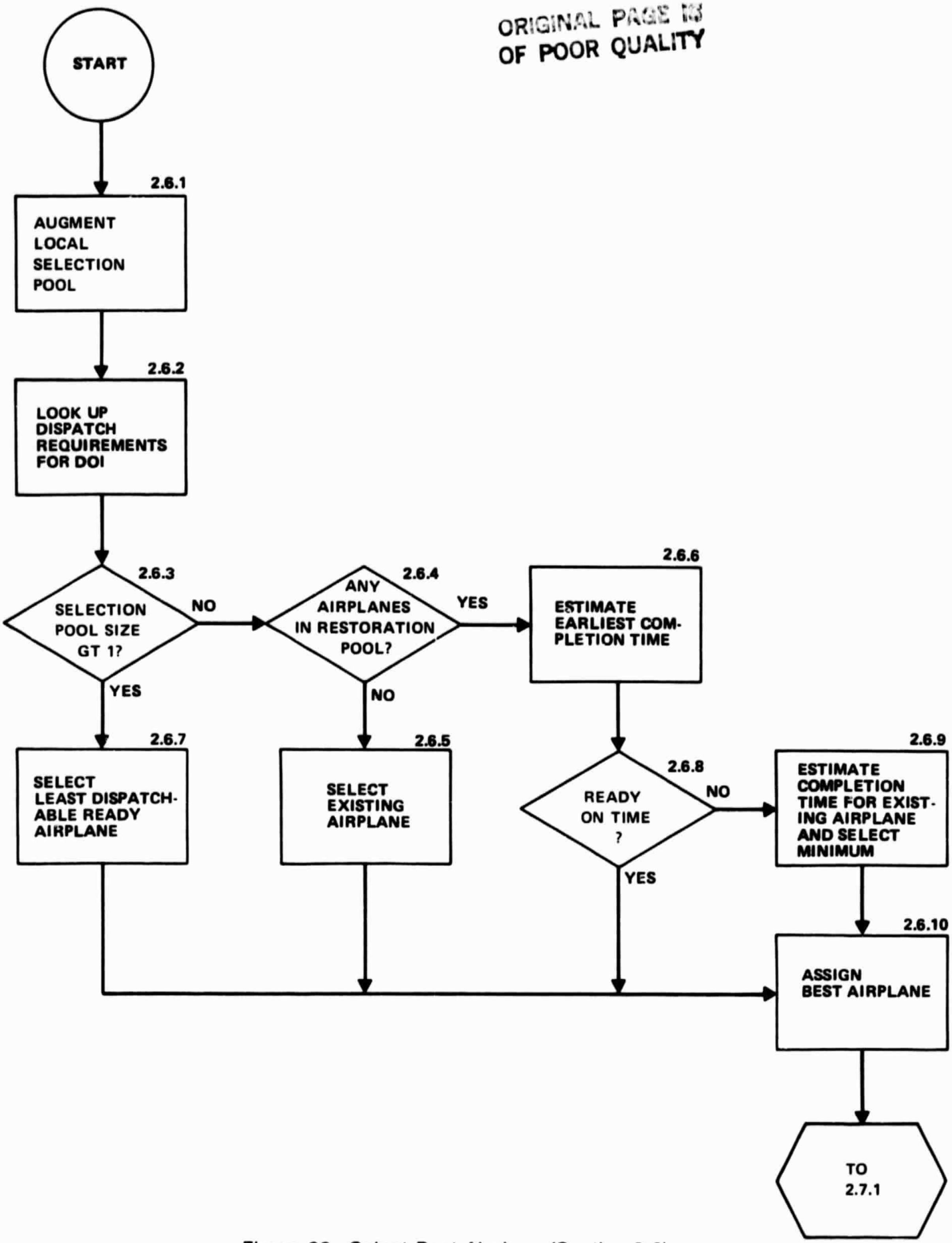


Figure 22. Select Best Airplane (Section 2.6)

Output

Additional airplane added to local selection pool

Processing

IF AIRPLANE IS VIRTUAL AND IF SPECIAL POOL IS NOT VACANT,
THEN DESTROY VIRTUAL AIRPLANE AND TRANSFER RESTORED AIRPLANE FROM SPECIAL POOL INTO LOCAL SELECTION POOL.
ELSE INCLUDE AIRPLANE IN LOCAL SELECTION POOL
ENDIF

NOTE: The special pool is composed of previously cancelled airplanes that have been restored to the most demanding fleet dispatch requirement.

2.6.2 Look Up Dispatch Requirements

Input

- Airplane model
- Flight number
- Minimum dispatch quantities by component stage: default values from availability file in system architecture data base, item 2.3; override values from line station data base, item 2

Output

- Table of minimum dispatch quantities for stages comprising the airplane model associated with the flight leg: to 2.6.7; also used in 2.8.2

Processing

- COPY TABLE OF MINIMUM STAGE DISPATCH QUANTITIES FOR SUBJECT AIRPLANE OF INTEREST AND FLIGHT NUMBER OF INTEREST

2.6.3 Is Size of Selection Pool GT 1?

NOTE: The selection pool comprises subject airplanes that are not irrevocably committed and are not being worked on.

Input

Quantity of airplanes in selection pool; internally generated via simulation utilities

Output

YES: directs flow to 2.6.7
NO: to 2.6.4

Processing

IF NUMBER OF AIRPLANES IN SELECTION POOL = 1
THEN NO
ELSE YES
ENDIF

2.6.4 Any Airplane in Restoration Pool?

Input

Size of restoration pool for airplane model of interest; see general remarks after 2.0

Output

YES: directs flow to 2.6.5
NO: to 2.6.7

Processing

IF SIZE OF RESTORATION POOL GREATER THAN ZERO
THEN YES
ELSE NO
ENDIF

2.6.5 Select Existing Airplane

Input

Existing airplane tail number: from 2.6.1

Output

- Existing airplane tail number: to 2.6.10
- Existing flag setting: to 2.11

Processing

GO TO 2.6.10

2.6.6 Estimate Earliest Completion Time

Input

For each airplane in the restoration pool of the same model that is required for the flight leg of interest:

- Tail number
- Time to go of all jobs underway

- Component status table
- Minimum dispatch requirements: via 2.6.2
- Expected restoration time for jobs not begun: from component data base, items 9.2 and 9.3

Output

- Earliest expected completion time and associated tail number: to 2.6.9

Processing

FOR EACH AIRPLANE IN RESTORATION POOL THAT IS NOT IRREVOCABLY ASSIGNED, ESTIMATE COMPLETION TIME OF JOBS IN PROGRESS THUS:

- TAKE LONGEST TIME TO GO OF ALL TASKS SIMULTANEOUSLY UNDERWAY AND ASSIGN THIS TO T1
- NEXT, COMPUTE THE TOTAL, DENOTED AS T2, OF EXPECTED JOB TIMES FOR TASKS NOT YET BEGUN AS FOLLOWS:
  - LRU(I,J) = LRU OF I'TH PART NUMBER AT JTH ADDRESS
    N(I,J) = QUANTITY OF LRU (I,J) BELOW DISPATCH MINIMUM
    Q(I,J) = QUANTITY OF NONFAILED LRU (I,J)
    D(I,J) = DISPATCH MINIMUM QUANTITY OF LRU (I,J)
    N(I,J) = D(I,J) - Q(I,J)
  - IF N(I,J) LT 0, SET N(I,J) = 0

    OVER THE RANGE OF ADDRESSES AND LRU ID NUMBERS, COMPUTE EXPECTED COMPLETION TIME, T2, AS

    (FIRST PIECE EXPECTED TIME (I) + (N(I,J)-1) * ADDITIONAL PIECE EXPECTED TIME (I))
  - ENDIF

  NOTE: This is an estimate using expected times, and assuming unstarted tasks are accomplished sequentially, without waiting or interruptions.

- COMPUTE THE MINIMUM OVER ALL THE TAIL NUMBERS OF (T1 + T2)
- CANDIDATE AIRPLANE HAS THE MINIMUM VALUE OF (T1 + T2)

2.6.7 Select Least Dispatchable Ready Airplane

Selecting the least dispatchable ready airplane for a particular flight increases overall dispatch reliability by saving the best airplanes for the most demanding flights.

Input

- Virtual flag settings for selection pool airplanes
- Component status tables for each eligible airplane in selection pool
- Dispatch minimum requirements for the flight leg: from 2.6.2

Output

- Tail number of least dispatchable airplane: to 2.6.10

Processing

```
FOR EACH AIRPLANE IN STATION'S SELECTION POOL COMPUTE A
DISPATCHABILITY INDEX IN THIS WAY:
      WHILE AIRPLANES REMAIN
         IF VIRTUAL FLAG IS SET
         THEN SET DISPATCHABILITY INDEX TO INFINITY
         ELSE
               FOR EACH FCS STAGE
               COMPUTE DIFFERENCE BETWEEN ACTUAL QUANTITY OF
               UNFAILED UNITS AND DISPATCH MINIMUM QUANTITY IF
               DIFFERENCE IS NEGATIVE, AIRPLANE IS UNDISPATCH-
               ABLE
               THEN CONSIDER NEXT AIRPLANE
               ELSE CONTINUE
               ENDIF
         SUM THESE DIFFERENCES OVER ALL STAGES TO YIELD
         DESIRED INDEX. SELECT AIRPLANE HAVING MINIMUM INDEX.
         NO TIE BREAKER REQUIRED.
         ENDIF
      ENDWHILE
```

2.6.8 Ready on Time?

Input

- Scheduled departure time for first flight leg of interest: from flight schedule data base, item 4

- Earliest completion time and tail number: from 2.6.6

- Irrevocable time limit for airplane assignments: default value from line station data base, item 1; or override value from item 16 of same data base

Output

- YES: directs flow to 2.6.10
- NO: to 2.6.9

C-2

Processing

- IF ESTIMATED COMPLETION TIME IS LE DEPARTURE TIME OF INTEREST MINUS IRREVOCABLE TIME LIMIT
  THEN YES
  ELSE NO
  ENDIF

2.6.9 Estimate Completion Time for Existing Airplane and Select Minimum of Expected Completion Times for Existing Airplane and Potential Substitute in Restoration Pool

Input

- Component status table for existing airplane: from 1.4, 1.8
- Minimum dispatch requirements for flight leg of interest: from 2.6.2
- Expected task times for mandatory LRU restorations: from component data base, items 9.2 and 9.3
- Minimum expected completion time and tail number of candidate airplane: from 2.6.6

Output

- Tail number of "best" airplane
- "Existing" flag setting: to 2.11

Processing

- REPEAT SECOND STEP IN 2.6.6 "PROCESSING" TO OBTAIN EXISTING AIRPLANE'S EXPECTED RESTORATION TIME

- IF T2 FOR EXISTING AIRPLANE LE (T1 + T2) FOR RESTORATION POOL'S CANDIDATE AIRPLANE

  THEN BEST AIRPLANE IS EXISTING AIRPLANE AND SET "EXISTING" FLAG

- ELSE CANDIDATE AIRPLANE FROM RESTORATION POOL IS BEST AIRPLANE

  ENDIF

2.6.10 Assign Best Airplane

Input

- Airplane tail number: from 2.6.5 or 2.6.7 or 2.6.9

Output

- Assignment of tail number to flight leg of interest

Processing

- ASSOCIATE TAIL NUMBER OF BEST AIRPLANE TO FLIGHT LEG OF INTEREST

2.7.1 Real Airplane?

Input

- Virtual flag setting via 1.1

Output

- YES: directs flow to 2.7.2
- NO: to 1.1

Processing

- IF VIRTUAL FLAG UNSET
  THEN YES
  ELSE NO
  ENDIF

2.7.2 Original Airplane?

Input

- Through-flight substitution tally
- Tail number of arriving airplane: from 2.1
- Tail number of best airplane: from 2.6.10
- Through-flight flag setting: from 7.2.5.3.3

Output

- YES: directs flow to 2.8
- NO: to 2.11 and 2.13
- Updated through-flight substitution tally

Processing

IF TAIL NUMBER OF ARRIVING AIRPLANE = TAIL NUMBER OF BEST AIRPLANE FOR FLIGHT LEG OF INTEREST
THEN YES
ELSE NO
    IF THROUGH-FLIGHT FLAG SET
    THEN INCREMENT THROUGH-FLIGHT SUBSTITUTION-TALLY BY 1
    ENDIF
ENDIF

2.8 Restore Airplane

This block is expanded in Figure 23 and the associated input-output-processing requirements are as follows:

2.8.1 Is Airplane at Line Maintenance Station?

Input

- Station maintenance capability code: from line station data base, item 13.1

Output

- YES: directs flow to 2.8.2
  NO: to 2.8.30

Processing

- IF MAINTENANCE CAPABILITY CODE = "M" (7.1.2, item 13.1)
  THEN YES
  ELSE NO
  ENDIF

2.8.2 Determine Mandatory and Deferrable Tasks

This block is expanded in Figure 24.

In addition to the subject FCS, line stations must serve the other systems comprising the subject airplane and also service other airplanes.
To produce a realistic environment for the subject FCS, the flight operations simulation generates failures and hence, line maintenance activity, for all the airline's flight equipment. For all systems other than the subject FCS, only dispatch critical failures that require mandatory maintenance are generated because these are the serious contenders with the subject FCS for line station labor. The CEM assumes that all line maintenance activity is performed by mechanics having the same skill level.

A growth version of the CEM could treat with two skill levels, avionic and mechanical, and generate separate queues.

2.8.2.1 Subject Airplane?

Input

- Subject airplane flag-passed via 2.20

Output

- YES: directs flow to 2.8.2.2
- NO: to 2.8.2.4

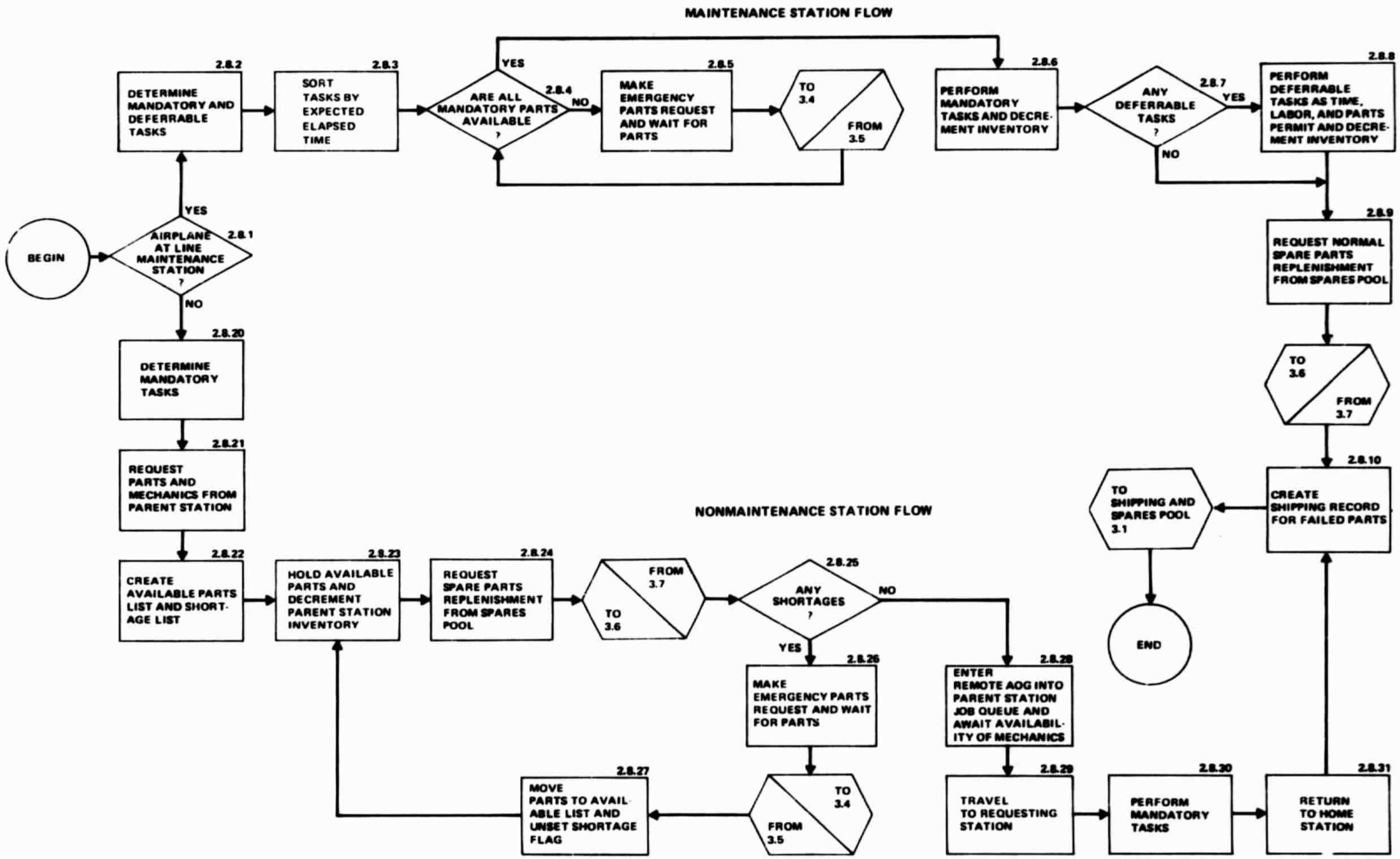


Figure 23. Restore Airplane to Dispatchability (Section 2.8)

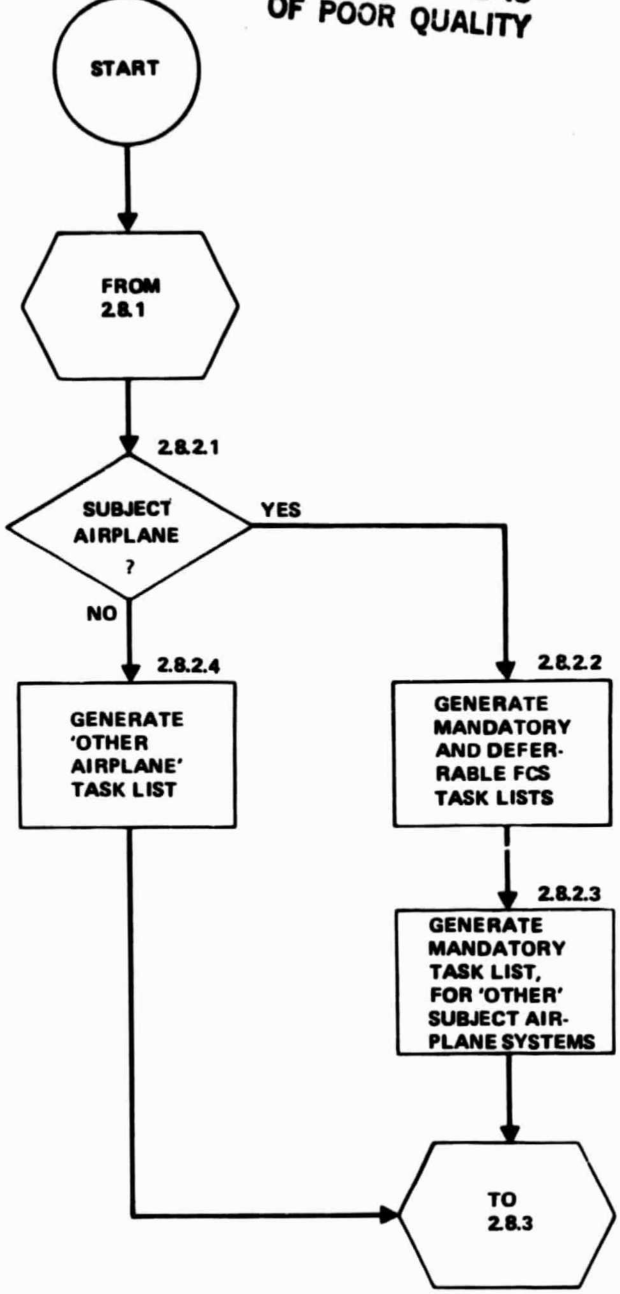


Figure 24. Determine Mandatory and Deferrable Tasks (Section 2.8.2)

Processing

IF "SUBJECT" AIRPLANE FLAG SET
THEN YES
ELSE NO
ENDIF

2.8.2.2 Generate Mandatory and Deferrable Task Lists

Input

- Component status table
- Minimum dispatch requirements via 2.6.2, or 2.14, or 2.18
- Inventory status table for line station from inventory status table, item 7.2.1
- Minimum dispatch quantities by stage: from 2.6.2
- Dependent effects table: from system architecture data base, item 3
- Virtual flag setting: from airplane status record, item 4.1
- Number of mechanics required: from component data base, item 10

Output

For each LRU part number, the task list record is as follows:

- Part number
- Mandatory restoration quantity and addresses
- Deferrable restoration quantity and addresses
- Number of mechanics required: used in 2.8.3, 2.8.4, 2.8.6

Processing

TO PROVIDE MAXIMUM CREATIVE FREEDOM TO THE IMPLEMENTOR, THIS SECTION IS PRESENTED IN DISCUSSION FORMAT. AN ILLUSTRATIVE EXAMPLE IS INCLUDED TO ILLUMINATE THE REQUIREMENT IN SUFFICIENT DETAIL TO EXPLICATE THE GENERAL PROBLEM THAT MUST BE TREATED.

Determination of Mandatory and Deferrable Tasks-FCS attributes include the familiar hierarchic organization of components, SRUs, and LRUs into tangible hardware entities (i.e., packages). In addition, fault-tolerant systems may involve performance criteria that span hardware boundaries, such as the minimum quantity of reconfigurable elements required for dispatch (e.g., the number of clocks in an Fault-Tolerant Multiprocessor [FTMP]).

Fundamental to this discussion is the concept of hardware addresses, which are nothing more than unique, symbolic locations for components. Consider an FTMP design that includes the following equipment complement:

QUANTITY	LRU PART NUMBER	NAME
5	201	I-line
5	202	O-line
5	203	C-line
5	204	P-line
10	407	LRU

The 10 number 407 LRUs are each comprised of these subunits:

	QUANTITY	PART NUMBER	NAME
NONRECONFIGURABLE	1	105	Power supply
	2	106	Bus guardian unit
	1	107	Bus interface unit
	1	108	Other
RECONFIGURABLE	1	109	CPU cache memory
	1	110	Memory
	1	111	Clock
	1	112	I/O port

The FTFCS, with minimum dispatch quantities, is illustrated in Figure 25. The addressing scheme is shown in Figure 26.

The minimum dispatchability criteria for the flight leg of interest are:

ADDRESS*	PART NUMBER	MINIMUM k	TOTAL COMPLEMENT AT THE ADDRESS n
1	201	3	5
1	202	3	5
1	203	4	5
1	204	5	5

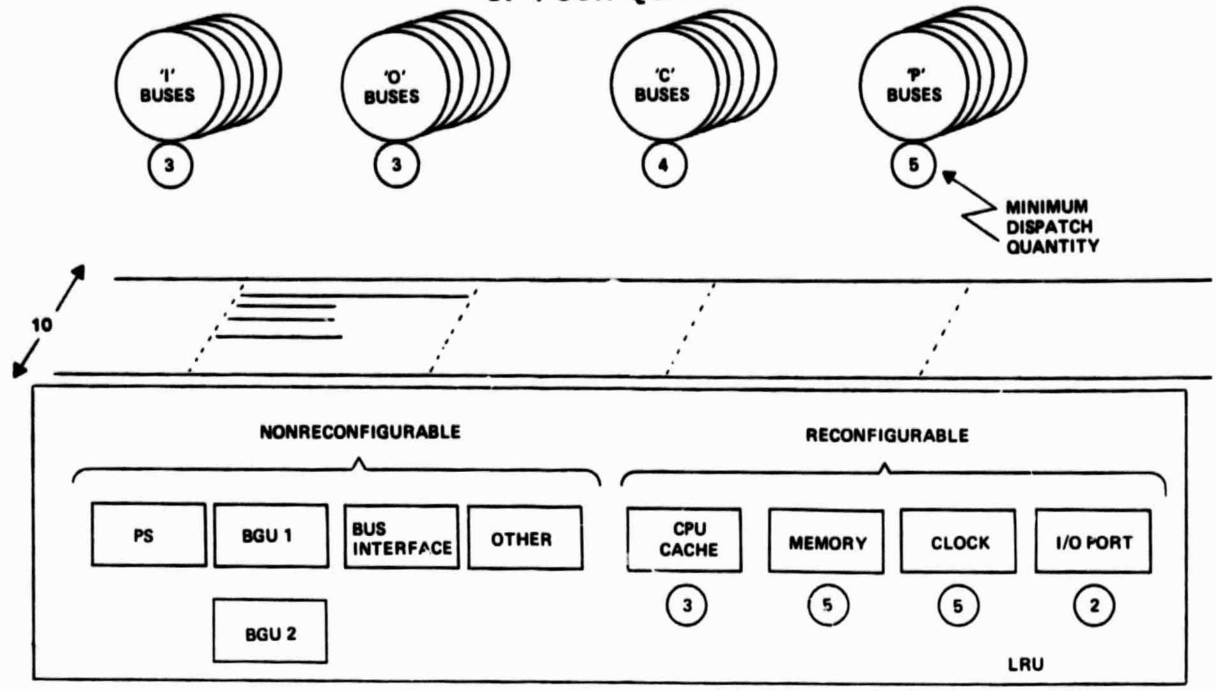


Figure 25. Hypothetical Flight Control Configuration

5 'I' LINES	A101	A102	A103	A104	A105
	LRU 201	LRU 201	LRU 201	LRU 201	LRU 201
5 'O' LINES	A106	A107	A108	A109	A110
	LRU 202	LRU 202	LRU 202	LRU 202	LRU 202
5 'C' LINES	A111	A112	A113	A114	A115
	LRU 203	LRU 203	LRU 203	LRU 203	LRU 203
5 'P' LINES	A116	A117	A118	A119	A120
	LRU 204	LRU 204	LRU 204	LRU 204	LRU 204

10 LRU 407'S

LRU 407								
POWER SUPPLY A35.1	BUS GUARDIAN UNIT A35.2	BUS GUARDIAN UNIT A35.3	BUS INTERFACE UNIT A35.4	OTHER A35.5	CPU CACHE A35.6	MEMORY A35.7	CLOCK A35.8	I/O PORT A35.9
LRU 105	LRU 106	LRU 106	LRU 107	LRU 108	LRU 109	LRU 110	LRU 111	LRU 112
⋮				⋮				⋮

LRU 407								
A44.1	A44.2	A44.3	A44.4	A44.5	A44.6	A44.7	A44.8	A44.9
LRU 105	LRU 106	LRU 106	LRU 107	LRU 108	LRU 109	LRU 110	LRU 111	LRU 112

Figure 26. Component Addresses

A multitude of choices may be made about what set of maintenance actions is the minimum necessary to restore the airplane to dispatchability. The real world algorithm for accomplishing this would be:

- Remove and replace failed LRUs having the greatest impact on dispatchability.
- Remove and replace at the component hierarchy level 3. (This excludes nullified LRUs; that is, unfailed LRUs rendered inoperative by the failure of other equipment.)
- Continue applying the first two rules until minimum dispatch conditions are met.

This has a heuristic aspect that is difficult to mechanize. Instead, the following set of rules will be employed:

- Restore the system at the hierarchy level 3 unless SRUs have failed, in which case restore at level 2.
- When given a choice, always restore a component the failure of which nullifies other dispatch-critical hardware.
- Continue until minimum dispatch conditions are met.
- If parts are out of stock, determine if there is an alternative set of maintenance actions to produce minimum dispatchability. With parts outages, the best set of maintenance actions is that which minimizes the quantity of spares that must be obtained on an emergency basis from the parts depot.

Applying the above rules to FTMP-1 results in these maintenance actions:

- Restore the P-line at address A119.
- Restore the LRU at address A35 (because the failed power supply is an SRU, the only recourse is to remove and replace at level 2).
- Restore the LRU at address A36 (similar logic as above).

For FTMP-2:

- Restore the P-line at address A119.
- Restore the power supply at address A35.1.
- Restore the bus guardian unit at address A36.2 and the clock at A36.8.

In addition to the subject FCS, line station responsibilities include restoring other airplane models comprising the fleet and other systems aboard the subject airplane. Addresses to be reserved are:

J.6 35 LE J LE 44	109	3	10
J.7 35 LE J LE 44	110	5	10
J.8 35 LE J LE 44	111	5	10
J.9 35 LE J LE 44	112	2	10

The next listing provides the relationship of independent failures to dependent events. Note that once component addresses are defined, dependent effects are fully described in terms of cause and effect addresses.

* Note: In Figure 26, addresses are prefixed by the letter A; however, use of this prefix is not necessary here.

DEPENDENT EFFECTS IF k-OF-n-FAIL

CAUSING ADDRESS	k	n	ADDRESSES AFFECTED	EFFECT CODE
35 LE J LE 44				
J.1	1	1	J.2 through J.9	D
J.2 through J.3	2	2	J.6, J.7, J.8, J.9	A
J.4	1	1	J.6, J.7, J.8, J.9	A
J.5	1	1	J.6, J.7, J.8, J.9	A

Effect Codes F: FAILED
A: NULLIFIED ENERGIZED
D: NULLIFIED UNENERGIZED

The components that are failed ultimately must be replaced. Those that are nullified are generally restored when the nullifying cause is corrected; however, if they are in an energized state while nullified they can still fail independently.

The components in cold standby are regarded as invulnerable to failures; when they are energized, they will assume not only the functions but also exchange addresses with the components they replace. Another attribute of k-of-n systems is that there may be nonunique ways of restoring them to service. A trivial case would be having to replace two units out of three failed. Which two? There also are other more complex considerations. They will be demonstrated by examining two variants, denoted FTMP-1 and FTMP-2, of the FTMP discussed previously; both have the same logical organization and the same failure patterns but differ in which units are line replaceable (LRU) and which units are shop replaceable (SRU), as shown in Figures 27 and 28.

ORIGINAL PAGE IS
OF POOR QUALITY

Lines	Addresses					Stage number	Quantity in service	Minimum dispatch quantity	Dispatchable?	LRU part number
I	A100 (F)					S2	4	3	Yes	201
O			A108 (F)	A109 (F)		S3	3	3	Yes	202
C		A112 (F)				S4	4	4	Yes	203
P				A119 (F)		S5	4	5	No	204

Legend:
(F) Failure
(N) Nullification
(Addresses containing no failures or nullifications are suppressed.)

LRU 407	SRU 105	SRU 106	SRU 106	SRU 107	SRU 108	SRU 109	SRU 110	SRU 111	SRU 112
Type 2 LRU	Power supply	Bus guardian unit	Bus guardian unit	Bus interface unit	Other	CPU cache	Memory	Clock	I/O port
A35	A35.1 (F)	STAGE S10 A35.2 (N)	A35.3 (N)	A35.4 (N)	A35.5 (N)	A35.6 (N)	A35.7 (N)	A35.8 (N)	A35.9 (N)
A36		STAGE S11 A36.2 (F)	A36.3 (F)			A36.6 (N)	A36.7 (N)	A36.8 (N)	A36.9 (N)
A37		STAGE S12 A37.2 (F)		A37.4 (F)		A37.6 (N)	A37.7 (N)	A37.8 (N)	A37.9 (N)
A38		STAGE S13					A38.7 (F)	A38.8 (F)	
A39		STAGE S14					A39.7 (F)		A39.9 (F)
A40		STAGE S15						A40.8 (F)	
A41		STAGE S16				A41.6 (F)			
A42		STAGE S17				A42.6 (F)			A42.9 (F)
A43		STAGE S18				A43.6 (F)	A43.7 (F)	A43.8 (F)	A43.9 (F)
A44		STAGE S19						A44.8 (F)	A44.9 (F)
	Stage number					S6	S7	S8	S9
	Quantity in service					4	4	3	3
	Minimum dispatch quantity					3	5	5	2
	Dispatchable ?					Yes	No	No	Yes

Figure 27. FTMP-1 (Mainly SRUs)

ORIGINAL PAGE IS OF POOR QUALITY

Lines	Addresses					Stage number	Quantity in service	Minimum dispatch quantity	Dispatchable?	LRU part number
I	A100 F					S2	4	3	Yes	201
O			A108 F	A109 F		S3	3	3	Yes	202
C		A112 F				S4	4	4	Yes	203
P				A119 F		S5	4	5	No	204

Legend:
F Failure
N Nullification
(Addresses containing no failures or nullifications are suppressed.)

LRU 407	LRU 105	LRU 106	LRU 106	SRU 107	LRU 108	SRU 109	SRU 110	LRU 111	SRU 112
Type 2 LRU	Power supply	Bus guardian unit	Bus guardian unit	Bus interface unit	Other	CPU cache	Memory	Clock	I/O port
A35	A35.1 F	STAGE S10 A35.2 N	A35.3 N	A35.4 N	A35.5 N	A35.6 N	A35.7 N	A35.8 N	A35.9 N
A36		STAGE S11 A36.2 F	A36.3 F			A36.6 N	A36.7 N	A36.8 N	A36.9 N
A37		STAGE S12 A37.2 F		A37.4 F		A37.6 N	A37.7 N	A37.8 N	A37.9 N
A38		STAGE S13					A38.7 F	A38.8 F	
A39		STAGE S14					A39.7 F		A39.9 F
A40		STAGE S15						A40.8 F	
A41		STAGE S16				A41.6 F			
A42		STAGE S17				A42.6 F			A42.9 F
A43		STAGE S18				A43.6 F	A43.7 F	A43.8 F	A43.9 F
A44		STAGE S19						A44.8 F	A44.9 F
Stage number						S6	S7	S8	S9
Quantity in service						4	4	3	3
Minimum dispatch quantity						3	5	5	2
Dispatchable ?						Yes	No	No	Yes

Figure 28. FTMP-2 (Mainly LRUs)

- ON THE SUBJECT AIRPLANE:

THE SUBJECT FCS	1
FCS COMPONENTS	26 and higher
OTHER, NON-FCS COMPONENT	4

- ON OTHER AIRPLANES (UP TO 10 TYPES) 6-15

The addressing convention has been employed in Figures 27 and 28 and Sections 2.8.2.3 and 2.8.2.4.

Stage numbers may be arbitrary with the exception that stage number S1 is reserved for the entire FCS. Stages are merely linked lists of addresses belonging to identical components. For instance, stage S8, a k-out-of-10 system (of clocks) is a collection of these address: A35.8, A36.8, ..., A44.8. In order to dispatch, five of these addresses must contain unfailed clocks. Another illustration would be stage S11, a 1-of-2 system of bus guardian units. When both units fail at addresses A36.2 and A36.3, the components at A36.6, A36.7, A36.8, and A36.9 are nullified.

Address A1 and part number 1 also are reserved for the entire FCS.

If an airplane is disassigned, the set of dispatch minimums to which it must be restored is the most demanding station requirement specified in the line station data base, item 18; cancelled airplanes must be restored to the most demanding fleet requirement, item 3 of the same data base.

In the case of the subject airplane's other components and the systems of other airplanes, the CEM will represent those systems via their unscheduled line removal rates for flight critical components. That is, only those failures requiring mandatory line maintenance will be generated. In this fashion, the subject FCS will contend for the labor resources at line stations.

2.8.2.3 Generate Mandatory Task List for "Other" Subject Airplane Systems

- Quantity of "other" non-FCS failures: from 1.5
- Line maintenance restoration time distribution type and parameters: from component data base, items 9.1, 9.2, 9.3
- Proportion of repair actions requiring two mechanics: from component data base, item 10

Output

- List of mandatory tasks for other avionics: to 2.8.3.3

Processing

- WHILE "OTHER" SUBJECT AIRPLANE FAILURES REMAIN
    - MAKE A DRAW FROM LINE MAINTENANCE RESTORATION TIME DISTRIBUTION
    - MAKE ANOTHER DRAW TO DETERMINE WHETHER ONE OR TWO

MECHANICS ARE REQUIRED

ENDWHILE

Create list with:

- LRU PART NUMBER (= 4 IN THIS CASE)
- MANDATORY RESTORATION QUANTITY (= 1)
- NUMBER OF MECHANICS REQUIRED

2.8.2.4 Generate "Other Airplane" Task List

Input

- Quantity of "other airplane" failures: from 1.3

- Line maintenance restoration time distribution type and parameters: from airplane data base, item 9.2

- Proportion of restoration actions requiring two mechanics: from component data base, item 10

Output

List of mandatory tasks for other airplane model of interest: to 2.8.3.5

Processing

- WHILE "OTHER AIRPLANE" FAILURES REMAIN

  - MAKE A DRAW FROM LINE MAINTENANCE RESTORATION TIME DISTRIBUTION

  - MAKE ANOTHER DRAW TO DETERMINE IF ONE OR TWO MECHANICS ARE REQUIRED

  ENDWHILE

NOTE: Each record in the list includes the following fields:

- LRU part number (permissible values are 6, 7..., 15)
- Mandatory restoration quantity
- Deferrable restoration quantity
- Number of mechanics required

2.8.3 Sort Tasks by Expected Elapsed Time

This block is expanded in Figure 29, and the associated input-output-processing follows:

2.8.3.1 Subject Airplane?

ORIGINAL PAGE IS
OF POOR QUALITY

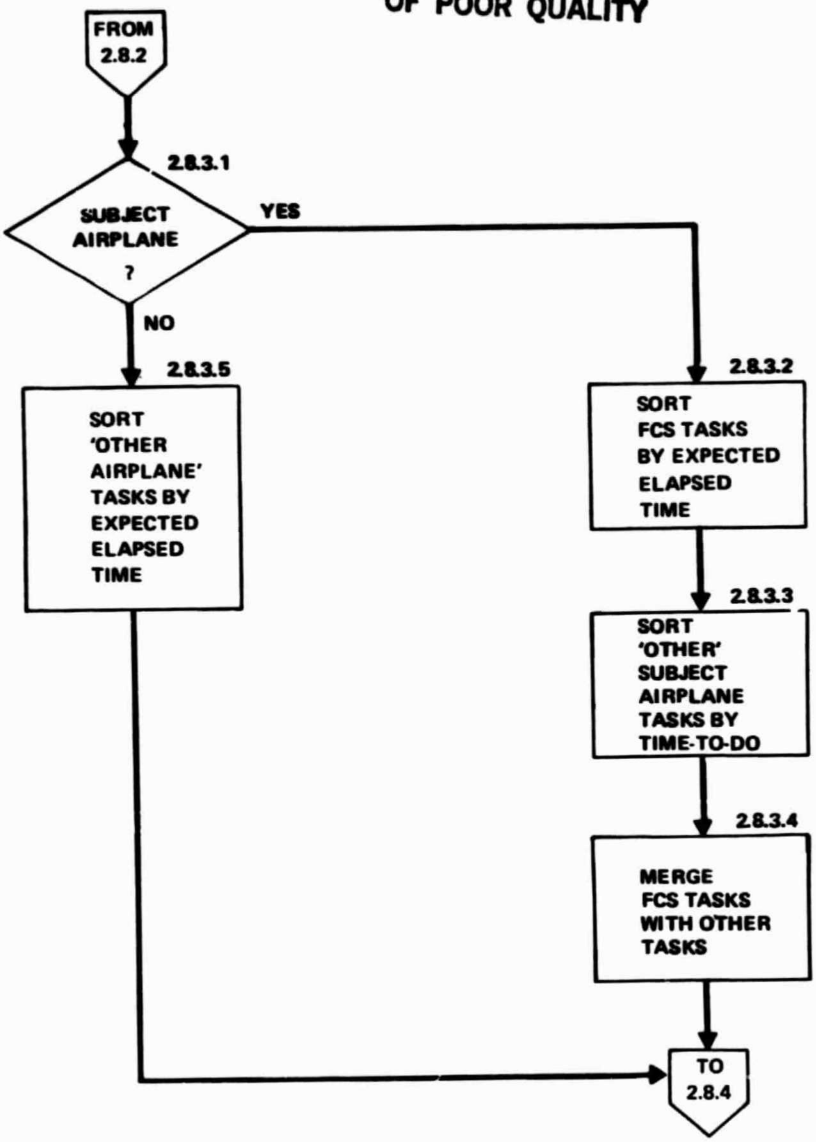


Figure 29. Sort Tasks by Expected Elapsed Time (Section 2.8.3)

Input

Subject airplane flag: passed via 2.20

Output

YES: directs flow to 2.8.3.2
NO: to 2.8.3.5

Processing

IF SUBJECT AIRPLANE FLAG SET
THEN YES
ELSE NO
ENDIF

2.8.3.2 Sort FCS Tasks by Expected Elapsed Time

Input

- Task list: from 2.8.2.2
  - LRU part number
  - Mandatory quantity to be restored
  - Deferrable quantity to be restored
  - Address

  Also, note that the task list has been screened for parts availability.

- Restoration times (first and additional) by part number from component data base, items 9.2 and 9.3

Output

- List of mandatory tasks sorted by number of mechanics required and expected elapsed time: to 2.8.3.4
- List of deferrable tasks sorted by number of mechanics required and expected elapsed time: to 2.8.7

Processing

- SEGREGATE TASKS INTO MANDATORY AND DEFERRABLE LISTS AND THEN BY NUMBER OF MECHANICS REQUIRED, YIELDING FOUR LISTS
- FOR EACH LIST, PERFORM THESE OPERATIONS:

  CALCULATE TASK TIME AS (RESTORATION TIME FOR FIRST ITEM) + (QUANTITY OF ITEMS TO BE RESTORED -1)* (RESTORATION TIME FOR ADDITIONAL ITEM)

- SORT EACH LIST BY TASK TIME IN ASCENDING ORDER

2.8.3.3 Sort "Other" Subject Airplane Tasks by Expected Elapsed Time

Input

List of mandatory tasks for other airplane type of interest: from 2.8.2.4

Output

Sorted list of mandatory tasks for subject airplane systems other than the FCS: to 2.8.3.4

Processing

- FOR ONE-MECHANIC TASKS, SORT IN ASCENDING ORDER BY EXPECTED ELAPSED TIME

- REPEAT FOR TWO-MECHANIC TASKS

2.8.3.4 Merge FCS Tasks With Other Tasks

Input

- Sorted one- and two-mechanic task lists from 2.8.3.2
- Sorted one- and two-mechanic task lists from 2.8.3.3

Output

- Merged one- and two-mechanic task lists: to 2.8.4, 2.8.5, 2.8.6

  - LRU part number
  - Number of mechanics
  - Quantity to be restored
  - Addresses

Processing

- MERGE MANDATORY ONE-MECHANIC TASK LISTS FOR FCS AND OTHER SUBJECT AIRPLANE SYSTEMS

- REPEAT FOR MANDATORY TWO-MECHANIC TASKS

2.8.3.5 Sort "Other Airplane" Tasks by Expected Elapsed Time

Input

List of mandatory tasks for airplane model of interest (other than subject airplane)

Output

Sorted list of mandatory line maintenance tasks for other (nonsubject) airplane

Processing

- FOR ONE-MECHANIC TASKS, SORT IN ASCENDING ORDER BY EXPECTED ELAPSED TIME
- REPEAT FOR TWO-MECHANIC TASKS

2.8.4 All Mandatory Parts Available?

Input

- Subject airplane flag: from airplane status record, item 3
- Mandatory task list: from 2.8.2.3
- Inventory status table

Note: Parts outages for the subject FCS, only, will be simulated.

Output

- YES: directs flow to 2.8.6
- NO: to 2.8.5
- LRU shortage list: to 2.8.5

Processing

```
IF SUBJECT AIRPLANE FLAG UNSET
THEN YES
ELSE
     FOR THE SUBJECT TAIL NUMBER
     WHILE FCS LRU PART NUMBERS REMAIN
        WHILE TASKS REMAIN
              SUM THE QUANTITIES REQUIRED TO RESTORE,
              YIELDING THE REQUIRED SUM
        ENDWHILE
        LET INVENTORY QUANTITY-REQUIRED SUM = X
        IF X LT ZERO
        THEN CREATE SHORTAGE RECORD WITH

              •    LRU PART NUMBER
              •    QUANTITY SHORT = ABSOLUTE VALUE OF X

        ENDIF
     ENDWHILE
     IF NO SHORTAGE RECORDS
     THEN YES
     ELSE NO
     ENDIF
ENDIF
```

NOTE: Non-FCS parts, in the CEM, are always available. The collection of shortage records comprises the shortage list.

2.8.5 Make Emergency Parts Request and Wait for Parts

Input

- LRU shortage list: from 2.8.4
- Time of request: from 2.3
- ID number of requesting station

Output

- Emergency quantities required by LRU part number: to 3.4

Processing

- REQUEST SPARES POOL TO SHIP EMERGENCY RESUPPLY TO LINE STATION IN QUANTITIES REQUIRED

- TIME-TAG THE REQUEST WITH COMPLETION TIME OF LAST OPERATION

2.8.6 Perform Mandatory Tasks and Decrement Local Inventory

Input

- Local inventory status table

- Sorted mandatory task list: from 2.8.3

- Component status table: latest update

- Dependent effects table: from system architecture data base, item 3

- Mechanic status table

- Total labor time accumulation for mandatory line maintenance by component part number

- Total overtime accumulation for the station

Output

- Updated inventory status table
- Updated component status table
- Time of completion: to 2.8.7 and 2.8.8
- Updated mechanic status table
- Decremented inventory status records for affected LRUs
- List of LRUs used by part number and quantity used
- Incremented total labor time accumulation by component part number
- Incremented overtime accumulation for the station

Processing

- PERFORM MANDATORY TASKS (RULES FOLLOW)

- DECREMENT INVENTORY AS FOLLOWS:
  - DECREMENT QUANTITY ON HAND BY QUANTITY REQUIRED FOR EACH PART NUMBER
  - DO THE ABOVE WHEN TASK BEGINS
- THE FOLLOWING RULES APPLY TO JOB SCHEDULING:

Rules for Scheduling Tasks at Line Maintenance Stations

1. These rules are invoked wherever an airplane requires mandatory maintenance; that is, deferrable maintenance will not be considered unless the airplane also requires mandatory maintenance.

2. Actual task times will be determined by appropriate random draws. The user may select from a variety of probability distributions.

3. The following priority scheme applies at line stations:

   P(2): Mandatory maintenance

   P(1): Deferrable tasks, including cancelled airplanes prior to cutoff time after which priority is raised to P(2). See 7.1.2, item 9.

   NOTE: A deferrable task, once undertaken, is raised to P(2) because the airplane cannot be dispatched with incomplete maintenance.

4. Available manpower is first scheduled to tasks of highest priority with preference then given to earliest scheduled departures.

5. No high priority, P(2), job in progress will be interrupted by lower priority work; however a P(2) job newly introduced into the job stream will interrupt other P(2) work if the interrupting airplane is assigned to a flight leg that is scheduled for earlier departure.

   NOTE: In the following rules, underlined quantities denote typical values.

6. When an interrupted job is resumed, its remaining time to go is increased by <u>0.25 hours</u>.

7. For two-person tasks. the first mechanic available waits for a second mechanic, unless the second is already working as a member of a two-person team, in which case the task will be performed by that two-person team when it completes its currrent assignment.

8. Intertask transit time is <u>10 minutes</u>.

9. Deferrable tasks may be undertaken on an airplane provided the expected delay does not increase (this enables parallel accomplishment of high- and low-priority tasks if labor is available that would otherwise be idle).

10. It is assumed that two or more jobs can proceed simultaneously on the same airplane.

11. No job will be interrupted if either of these conditions are met:

- If the job is at least 90% complete
- If the job will be completed in 15 minutes

12. All interrupted jobs on an airplane must be complete prior to its departure.

13. Remote AOGs will enter the job stream of the parent station.

14. Once begun, remote jobs are uninterruptible.

15. Virtual airplanes, which represent cancelled flights, will not require maintenance.

16. Postflight checkouts at maintenance line stations will be assigned high priority, P2, and will require one mechanic; at nonmaintenance line stations, this function requires flight crew members and therefore does not contend for mechanics.

17. To support a given airplane, the tasks are undertaken in this sequence:

- Short one-person mandatory tasks, P(2) priority
- Short two-person mandatory tasks, P(2) priority
- Long two-person mandatory tasks, P(2) priority
- Long one-person mandatory tasks, P(2) priority
- Short one-person tasks, P(1) priority
- Long one-person deferrable tasks, P(1) priority
- Short two-person deferrable tasks, P(1) priority
- Long two-person deferrable tasks, P(1) priority

Short tasks are defined as those tasks which have expected elapsed times of 0.75 hours or less.

18. Overtime will be accumulated. At shift end, mechanics will continue any tasks in progress until completion; the CEM will not model job handoffs. Any excess over the standard workday will be treated as overtime hours.

19. At maintenance-capable stations, inventory is decremented at the beginning of tasks; at nonmaintenance stations, parts are reserved when the emergency maintenance request is received at the parent station.

2.8.7 Any Deferrable Tasks?

Input

- Task list: from 2.8.2.2

Output

- YES: directs flow to 2.8.8
- NO: to 2.8.9

Processing

IF DEFERRABLE TASK LIST VACANT
THEN NO
ELSE YES
ENDIF

2.8.8 Perform Deferrable Tasks as Time, Labor, and Parts Permit and Decrement Inventory

Refer to processing subsection of 2.8.6 for ranking of deferrable tasks in job scheduling algorithms. Outputs are the same as for 2.8.6, with the exception that overtime tallies are incremented for deferrable line maintenance.

2.8.9 Request Spare Parts Replenishment From Spares Pool

Input

- Line station identification number
- List of parts and quantities used for mandatory maintenance: from 2.8.6
- List of parts and quantities used for deferrable maintenance: from 2.8.8
- Completion time of last performed maintenance task: from 2.8.6 or 2.8.8
- Desired stock level table: from shipping and inventory data base, item 3

Output

- Request list for parts replenishment: to 3.6
- Time of request: to 2.8.10 and 3.6

Processing

- MERGE LIST OF PARTS USED FOR MANDATORY AND DEFERRABLE MAINTENANCE

- WHILE PART NUMBERS REMAIN ON MERGED LIST
  IF DESIRES STOCK LEVEL AT LINE STATION IS NOT ZERO
  THEN WRITE PARTS REQUEST RECORD WITH COMPONENT PART NUMBER AND QUANTITY REQUIRED,
  ELSE CONTINUE
  ENDIF
  ENDWHILE

- TIME-TAG THE LIST WITH COMPLETION TIME OF LAST PERFORMED LINE MAINTENANCE TASK (THIS IS THE TIME OF THE SPARES REQUEST)

- IDENTIFY THE LIST WITH ORIGINATING STATION'S ID NUMBER

2.8.10 Create Shipping Record for Failed Parts

Input

- Component repair code: from component data base, item 4.1
- Time of spares request: from 2.8.9

Output

- Shipping code setting that indicates whether discrepant part is shipped to repair shop or directly to vendor: used by shipping and spares pool, 3.1

Processing

- IF REPAIR CODE = "VV"
  THEN SET SHIPPING CODE = 2
  ELSE SET SHIPPING CODE = 1

  NOTE: Repair codes are:

  AA: Airline test and airline repair
  AV: Airline test and vendor repair
  AT: Airline test and throwaway
  VV: Vendor test and vendor repair

- SET TIME OF SHIPMENT = TIME OF SPARES REQUEST

- CREATE SHIPPING RECORD FOR EACH COMPONENT PART NUMBER WITH THESE FIELDS:

  - COMPONENT PART NUMBER
  - TIME OF SHIPMENT
  - SHIPPING CODE
  - QUANTITY SHIPPED

2.8.11 Through 2.8.19

These numbers are unused.

2.8.20 Determine Mandatory Tasks

Input

- Replace minimum dispatch requirements in 2.8.2.2 input list with most demanding minimum dispatch levels: from line station data base, item 3

Output

- Same as 2.8.2.2: outputs are used in 2.8.21 through 2.8.29

Processing

- THE SAME AS 2.8.2.2

  NOTE: If an unfailed time-monitored component has an "OVER-LIMIT" flag set (see 1.6.5), its replacement is deferred until its airplane arrives at a maintenance-capable station.

2.8.21 Request Parts and Mechanics From Parent Station

Input

- Task list records from 2.8.20

  Recall that the record structure includes these fields for each task:

  - LRU part number
  - Quantity to be restored
  - Addresses
  - Number of mechanics required

- Most demanding airline dispatch requirement table: from line station data base, item 3

- Parent station ID: from line station data base, item 13.2

Output

- Quantity of mechanics required: to 2.23

- List of spare parts required by LRU part number and quantity: to 2.8.23, 2.8.24, and 2.8.25

Processing

```
UNSET "TWO MECHANICS" FLAG
WHILE LRU PART NUMBERS REMAIN
      WHILE TASKS REMAIN
            SUM MANDATORY QUANTITIES
            IF REQUIRED NUMBER OF MECHANICS = 2
            THEN SET "TWO MECHANICS" FLAG
            ENDIF
      ENDWHILE

      REQUEST PARTS FROM PARENT STATION INVENTORY
ENDWHILE
IF "TWO MECHANICS" FLAG SET,
THEN REQUEST TWO MECHANICS FROM PARENT STATION
ELSE REQUEST ONE MECHANIC
ENDIF
```

2.8.22 Create Available Parts List and Shortage List

Input

- Parent station ID: from line station data base, item 13.2
- Parts request list: from 2.8.21
- Inventory status table (7.2.1) at parent station

Output

- Available parts list: to 2.8.23
- Shortage list: to 2.8.23
- Shortage flag setting: to 2.8.25

Processing

```
LET INVENTORY QUANTITY AT PARENT STATION—REQUIRED QUAN-
TITY = X
WHILE LRU ID NUMBERS REMAIN
      IF X LT ZERO
      THEN  CREATE SHORTAGE RECORD WITH

      • LRU PART NUMBER
      • QUANTITY SHORT

      ELSE  CREATE AVAILABILITY RECORD WITH

      • LRU PART NUMBER
      • QUANTITY REQUIRED

      ENDIF
ENDWHILE
IF SHORTAGE RECORDS EXIST
THEN  SET SHORTAGE FLAG
ELSE  CONTINUE
ENDIF
```

2.8.23 Hold Available Parts and Decrement Parent Station Inventory

NOTE: Because of the seriousness of AOG situations at nonmaintenance line stations, this block reserves the parts upon receipt of the request, contrary to the practice at maintenance stations, which decrement inventory only upon initiating a task.

Inputs

- Availability parts list: from 2.8.22
- Inventory status table for parent station
- Arrival time at line station: from 1.8
- Arrival time of emergency parts at parent station: from 3.5

Outputs

- Decremented inventory status records at parent station

Processing

- FOR EACH RECORD IN AVAILABLE PARTS LIST, REDUCE QUANTITY ON HAND BY QUANTITY OF PARTS REQUIRED.

2.8.24 Request Spare Parts Replenishment From Spares Pool

Input

- Line station ID number
- Parent station ID number: from line station data base, item 13.2
- Available parts list: from 2.8.22
- Completion time of postflight check: from 2.3
- Desired stock level table: from shipping and spares data base, item 3

Output

- Parts request list: to 3.6

Processing

- IDENTIFY THE LIST WITH ID OF ORIGINATING STATION

- WHILE PART NUMBERS REMAIN ON AVAILABLE PARTS LIST
      WRITE PARTS REQUEST RECORD WITH COMPONENT PART NUMBER AND QUANTITY REQUIRED
  ENDWHILE

2.8.25 Any Shortages?

Input

- Shortage flag setting: from 2.8.22

Output

- YES: directs flow to 2.8.26
- NO: to 2.8.28

Processing

```
IF SHORTAGE FLAG SET
THEN YES
ELSE NO
ENDIF
```

2.8.26 Make Emergency Parts Request and Wait for Parts

Inputs

- LRU shortage list: from 2.8.22
- Time of request: from 2.3

Output

- Expected arrival time of emergency resupply: to 2.8.6
- Emergency request list by LRU part number and quantity required: to 3.4

Processing

- REQUEST SPARES POOL TO SHIP EMERGENCY RESUPPLY TO PARENT LINE STATION IN QUANTITIES REQUIRED

- AFFIX TIME TAG WITH TIME = COMPLETION TIME OF POSTFLIGHT CHECK

2.8.27 Move Parts to Available List and Unset Shortage Flag

Input

- Shortage flag setting: from 2.8.25
- Shortage list: from 2.8.22
- Available parts list: from 2.8.22

Output

- New available parts list: to 2.8.23

Processing

- CONVERT SHORTAGE LIST TO AVAILABLE LIST
- UNSET SHORTAGE FLAG

2.8.28 Enter Remote AOG Into Parent Station's Job Queue and Await Availability of Mechanics

Input

- See 2.8.6

- In addition, minimum depletion level for mechanics at parent station: from line station data base, item 14.3

- Mechanic status table for parent station

Output

- Time at which mechanics are available for travel to remote site: to 2.8.26

Processing

- SAME AS 2.8.6

  NOTE: Mechanics cannot be dispatched if doing so results in station going below level specified in line station data base 7.1.2, item 14.3.

2.8.29 Travel to Requesting Station

Input

- Time at which mechanics are available: from 2.8.25
- Travel time from parent station: from line station data base, item 13.3
- Mechanic status table associated with home station

Output

- Time at the beginning of emergency maintenance
- Updated mechanic status table at parent station

Processing

- TIME AT WHICH EMERGENCY MAINTENANCE BEGINS = AVAILABILITY TIME OF MECHANICS + TRAVEL TIME FROM PARENT STATION

- UPDATE MECHANIC STATUS TABLE AT HOME STATION

2.8.30 Perform Mandatory Tasks

Input

- Mandatory task list: from 2.8.20
- Total labor spent on emergency maintenance, overtime and regular

Output

- Updated component status table
- Completion time: to 2.8.9
- Increments to regular and overtime hour accumulations

Processing

- SAME AS 2.8.6 EXCEPT RULE 9 AND 19
- FLAG ALL THE AOG TASKS AS UNINTERRUPTABLE
- INCREMENT OVERTIME ACCUMULATIONS
- INCREMENT REGULAR TIME ACCUMULATIONS

2.8.31 Return to Home Station

Input

- Completion time: from 2.8.27

- Number of mechanics
- Travel time to home station: from line station data base, item 13.4
- Mechanic status table

Output

- Updated mechanic status table

Processing

- ARRIVAL TIME AT HOME STATION
  = COMPLETION TIME + TRAVEL TIME TO HOME STATION

2.9 Perform Preflight Check

Input

- Preflight checkout time: from airplane data base, item 3
- Start time of preflight checkout: internally generated

Output

- Completion time of preflight checkout

Processing

- ADVANCE SIMULATION CLOCK BY PREFLIGHT CHECKOUT TIME

2.10 Still Dispatchable to Flight Leg of Interest

NOTE: At this time, some failures not detected at the previous postflight check become visible.

Input

- See 2.5

Output

- YES: directs flow to 2.15
- NO: to 2.6

Processing

- SEE 2.5

2.11 Is Dispatchable Airplane From Selection Pool or Restoration Pool?

Input

- "Existing" flag setting: from 2.6.5 or 2.6.9

Output

- YES: directs flow to 2.9
- NO: to 2.12

Processing

IF "EXISTING" FLAG SET
THEN NO
ELSE YES
ENDIF

2.12 Wait for Restoration Completion

Input

- Event notice that assigned airplane's restoration is complete, with time of completion: generated in 2.14

Output

- Time of restoration completion: to 2.9

Processing

- ADVANCE CLOCK TO TIME AT WHICH RESTORATION IS COMPLETED

2.13 This block is deleted

2.14 Restore Airplane

Input

- Most demanding station dispatch requirements: from line station data base, item 3
- For other inputs, outputs, and processing, see 2.8

Output

- See 2.8

Processing

- SEE 2.8
- IN ADDITION, SET ALL FAILURE VISIBILITY CODES IN FCS OF INTEREST TO "KNOWN"

2.15 Can Airplane Depart on Time?

Input

- Time at completion of preflight check: from 2.9
- Flight number associated with airplane: from 2.2 or 2.6
- Scheduled departure time for DOI: from flight schedule data base, item 4

Output

- YES: directs flow to 1.1
- NO: to 2.16

Processing

IF SCHEDULED DEPARTURE TIME GE
TIME AT SUCCESSFUL COMPLETION OF PREFLIGHT CHECK
THEN YES
ELSE NO
ENDIF

2.16 Accumulate Delay Time

Input

- Scheduled departure time: from flight schedule data base, via 2.15

- Time at completion of preflight check: from 2.9 via 2.15

- Previous delay time total for affected airplane model: from latest access to this block

- Minimum delay time resulting in a cancellation: from flight schedule data base, item 7

Output

- Updated delay time accumulation for affected airplane model

Processing

- DELAY TIME INCREMENT = THE LESSER OF THE FOLLOWING:

    - MINIMUM DELAY TIME RESULTING IN A CANCELLATION FOR DOI

    - TIME AT COMPLETION OF PREFLIGHT CHECK MINUS SCHEDULED DEPARTURE TIME

2.17 Flight Cancelled?

Input

- Time at completion of preflight check: from 2.9 via 2.16
- Scheduled departure time: from flight schedule data base via 2.15
- Delay time resulting in a cancellation, via 2.16

Output

- YES: directs flow to 2.18
- NO: to 2.20

Processing

IF TIME AT COMPLETION OF SUCCESSFUL PREFLIGHT CHECK LE (SCHEDULED DEPARTURE TIME + DELAY TIME RESULTING IN A CANCELLATION)
THEN YES
ELSE NO
ENDIF

2.18 Restore Airplane to Most Demanding Fleet Requirement and Increment Special Pool

Input

- Table of most demanding fleet requirements (used in 2.8.2): from line station data base, item 3

- Average relocation time for cancelled airplanes: from line station data base, item 6

- Local cutoff time after which cancelled airplanes are increased in priority to priority (2): from line station data base, item 9

- For other input, see 2.8

Output

- Incremented special pool
- For other output, see 2.8
- Completion time

Processing

- ASSIGN PRIORITY (2) TO AIRPLANE IF TIME GE LOCAL CUTOFF TIME

- INCREMENT SPECIAL POOL

- FOR OTHER PROCESSING SEE 2.8

- AFTER COMPLETING RESTORATION OF AIRPLANE TO MOST DEMANDING FLEET REQUIREMENTS, AIRPLANE ENTERS SPECIAL POOL AT TIME = RESTORATION COMPLETION TIME + AVERAGE RELOCATION TIME FOR CANCELLED AIRPLANES

2.19 Assign Virtual Airplane to Cancelled Airplane's Flight Leg

Input

- DOI of cancelled airplane: from 2.2 or 2.4
- Flight code: from flight schedule data base, item 6

Output

- Association of DOI with virtual airplane
- Updated revenue cancelled flight tally
- Updated nonrevenue cancelled flight tally

Processing

- ASSIGN VIRTUAL AIRPLANE TO CANCELLED AIRPLANE'S DOI

- SET VIRTUAL AIRPLANE FLAG

- IF FLIGHT CODE = "N" (sec. 7.1.7; item 6)
  THEN INCREASE NONREVENUE CANCELLED FLIGHT TALLY BY 1
  ELSE INCREASE REVENUE CANCELLED FLIGHT TALLY BY 1
  ENDIF

2.20 Schedule a Departure

Input

- Subject airplane flag: from airplane status record, item 3
- Virtual flag setting: from airplane status record, item 4.1, updated in 2.19
- Completion time of postflight checkout: internally generated
- Flight number from 2.4
- Scheduled departure time: via 2.4 from flight schedule data base, item 4

Output

Event notice with:

- Airplane model
- Airplane tail number
- Flight number
- Departure time
- Subject airplane flag setting
- Virtual airplane flag setting
- Through flight flag setting

Output is used by Flight Operations, 1.1

Processing

ISSUE AN EVENT NOTICE WITH

- AIRPLANE MODEL
- AIRPLANE TAIL NUMBER
- FLIGHT NUMBER
- ACTUAL DEPARTURE TIME
- SUBJECT AIRPLANE FLAG SETTING
- VIRTUAL FLAG SETTING
- THROUGH FLIGHT FLAG SETTING

IF FLIGHT NOT CANCELLED
THEN DEPARTURE TIME IS THE GREATER OF

- COMPLETION TIME OF SUCCESSFUL PREFLIGHT CHECK
- SCHEDULED DEPARTURE TIME

ELSE DEPARTURE TIME OF VIRTUAL FLIGHT = SCHEDULE DEPARTURE TIME
ENDIF
IF VIRTUAL FLAG SET
THEN TALLY A VIRTUAL FLIGHT FOR AIRPLANE MODEL OF INTEREST
ELSE TALLY A REGULAR FLIGHT FOR AIRPLANE MODEL OF INTEREST
ENDIF

5.3 SHIPPING AND INVENTORY MANAGEMENT SIMULATION

Primary Functions. This module has two functions:

- Generates time lags for shipping actions
- Maintains LRU inventories at line stations and the parts depot

The parts hierarchy defined in the FCS system architecture data base will be restated.

- Type 1: an LRU that has no subunits and is not contained within any other LRU

- Type 2: an LRU that contains subunits

- Type 3: an LRU that is a subunit of a type 2 LRU

- Type 4: an SRU that is a subunit of a type 2 LRU

Shipping Actions. The shipping actions processed by this module are summarized in Table 2.

Table 2. Shipping Activity Summary

SHIPPING CODE	REFERENCE	SHIPPING ACTION FROM	SHIPPING ACTION TO	COMPONENT CONDITION	COMPONENT TYPES	NOTES*
1	2.8.10	Line station	Repair shop	Failed	1, 2, 3	
2	2.8.10	Line station	Vendor	Failed	1, 2, 3	Repair code VV,
3	6.36	Repair shop	Parts depot	Repaired	1, 2, 3	
4	6.36	Repair shop	Vendor	Failed	1, 2, 3, 4	Repair code AV,
5	5.2	Vendor	Repair shop	Repaired, (but needing repair confirmation)	1, 2, 3, 4	Repair code AV,
6	5.2	Vendor	Parts depot	Repaired	1, 2, 3	Repair code VV,
7	4.2	Hangar	Repair shop	Failed	1, 2, 3	
8	4.2	Hangar	Vendor	Failed	1, 2, 3	Repair code VV,
9	3.3	Parts depot	Line station	Repaired	1, 2, 3	
10	3.3	Parts depot	Hangar	Repaired	1, 2, 3	

* Repair codes are provided in Secton 2.8.10.

Inventory Management. Throwaway components: The CEM assumes that throwaway components for line and shop are in plentiful supply whenever needed. The simulation tallies the demand for throwaways for conversion into cost accumulations.

SRUs and type 3 LRUs in the repair shop: SRUs and type 3 LRUs in the repair shop also will not be subject to outages. Nonthrowaway type 3 LRUs are recirculated within the repair shop, and its initial stock is constantly replenished. Therefore, inventory management is not required in this instance beyond setting the initial allocation, which is self-perpetuating thereafter. The recommended approach is to initially allocate an excessive quantity of type 3 and 4 components (sub-LRUs and SRUs) to the repair shop, monitor the lowest level reached during a run, and subtract this minimum quantity from the initial excessive quantity to yield the final allocation. Otherwise, the simulation will have to create new parts whenever an unsatisfied demand occurs, an awkward process with time-monitored components that have unique serial numbers.

Type 1, 2, and 3 LRUs in the hangar: The scheduled maintenance facility (hangar) performs its functions in zero time (sec. 4.0). Parts shortages that would create additional hangar delay are not permitted in the CEM. Therefore, the adjusted excessive allocation technique cited above also will be used for the hangar.

Type 1 and 2 LRUs at line stations and parts depot: Inventory allocations based upon expected demand are calculated in the AOM portion of the CBOM, and these levels are maintained via simple replenishment algorithms in 3.5 and 3.7. A future growth direction for the CEM will be to duplicate the calculations of the AOM every time components are shipped. In the event of failure to meet or closely approximate the assumptions underlying initial inventory allocation of the AOM, such an algorithm would adjust the initial allocations dynamically.

Attrition costs are handled outside the simulation by multiplying the annual number of a component's repair actions by the attrition rate (expressed in attritions per repair action) and the unit replacement cost.

Logic for the shipping and spares pool simulation, shown in Figure 30, follows:

3.1 Repaired Parts?

Input

- Shipping codes: written into component status record in 2.8.11, 4.2, 5.2, or 6.35

Output

- YES: directs flow to 3.2
- NO: to 3.4

Processing

- IF SHIPPING CODE = 1 OR 2 OR 4 OR 7 OR 8
- THEN NO

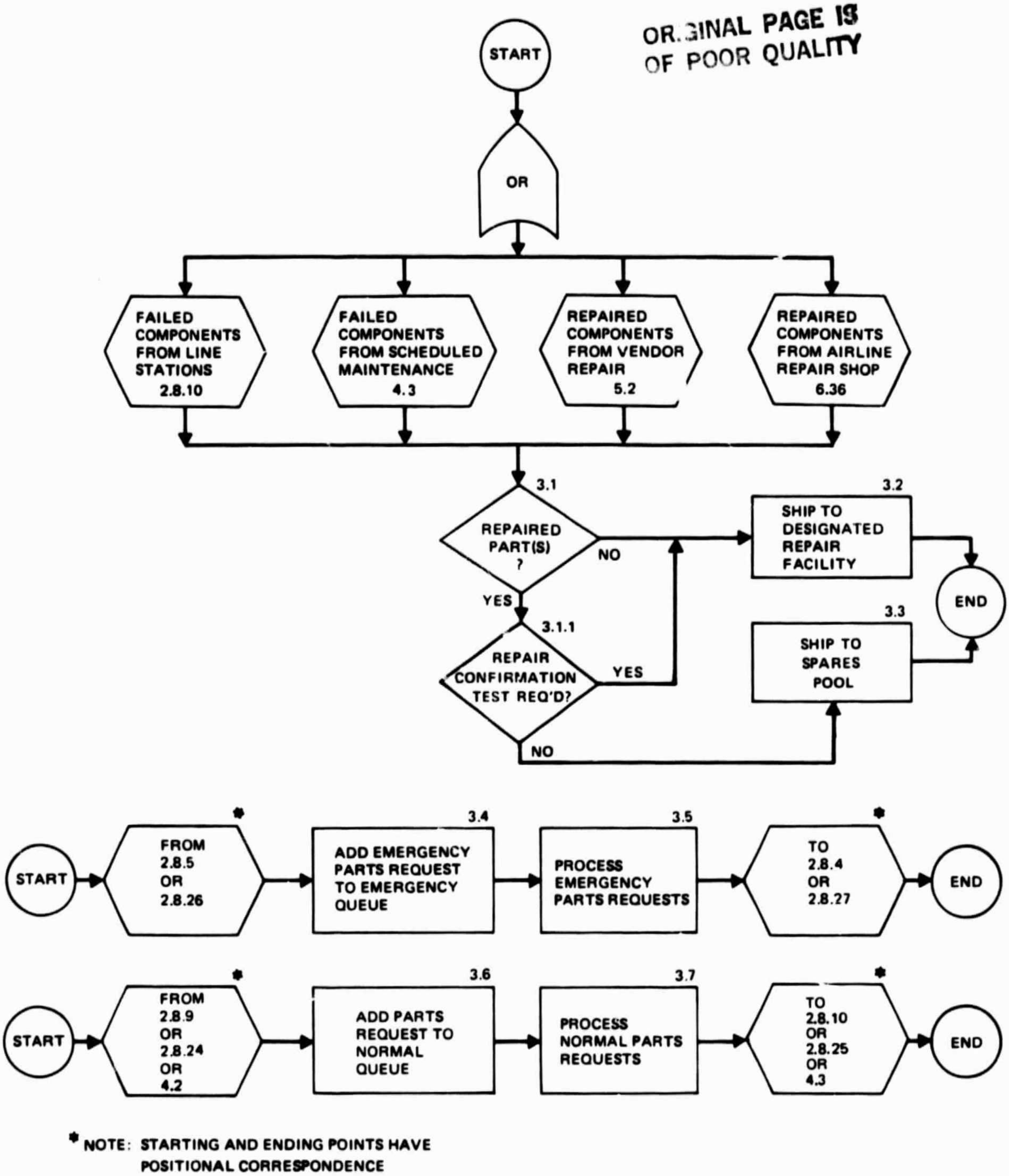


Figure 30. Shipping and Spares Pool (Section 3.0)

- ELSE YES
- ENDIF

3.1.1 Repair Confirmation Test Required?

NOTE: Parts that are tested by the airline, sent to a vendor for repair, and then retested by the airline to confirm the repair will have repair code "AV" and will be assigned a shipping code "6" in block 5.2.

Input

Shipping code: from third field in shipping record created in 5.2

Output

- YES: directs flow to 3.2
- NO: to 3.3

Processing

NOTE: Shipping code is either 5 or 6
IF SHIPPING CODE = 6
THEN YES
ELSE NO

3.2 Ship to Designated Repair Facility

Input

- Shipping codes: passed via 3.1
- Shipping time to vendor (if applicable): from component data base, item 4.4
- Shipping data base: all items

Outputs

- Destination arrival times

Processing

CASE I: SHIPPING CODE = 1 (LINE STATION TO REPAIR SHOP)
ADVANCE CLOCK BY SHIPPING TIME TO REPAIR STATION FOR MAINTENANCE LINE STATION OF INTEREST

CASE II: SHIPPING CODE = 2 (LINE STATION TO VENDOR)
ADVANCE CLOCK BY SHIPPING TIME TO VENDOR FROM LINE STATION

CASE III: SHIPPING CODE = 4 (REPAIR SHOP TO VENDOR)
ADVANCE CLOCK BY SHIPPING TIME TO VENDOR FROM REPAIR SHOP

CASE IV: SHIPPING CODE = 7 (HANGAR TO REPAIR SHOP)
ADVANCE CLOCK BY SHIPPING TIME FROM HANGAR TO REPAIR SHOP

CASE V: SHIPPING CODE = 8 (HANGAR TO VENDOR)
ADVANCE CLOCK BY SHIPPING TIME FROM HANGAR TO VENDOR

3.3 Ship to Spares Pool

Input

- Shipping record: from 5.2 or 6.36
- Inventory status table for spares pool

Output

- Incremented inventory status record for part of interest

Processing

- INCREMENT SPARES QUANTITY ON HAND BY 1 FOR SAME PART NUMBER AS THAT IN THE SHIPPING RECORD

- ARRIVAL TIME OF PARTS = TIME OF SHIPMENT + SHIPPING TIME

3.4 Add Emergency Parts Request to Emergency Queue

Input

- Time of request: from 2.8.5 or 2.8.26

- ID of requesting station

- Request list by component part number and quantity required: from 2.8.5 or 2.8.26

Output

Incremented emergency first-in-first-out (FIFO) queue: to 3.5.1

Processing

APPEND EMERGENCY PARTS REQUEST LIST TO FIFO EMERGENCY QUEUE

3.5 Process Emergency Parts Request

This block is expanded in Figure 31.

This block enables the spares pool to ship parts on an expedited basis. If the parts depot is out of certain parts, the parts in the repair cycle are raised in priority

ORIGINAL PAGE IS
OF POOR QUALITY

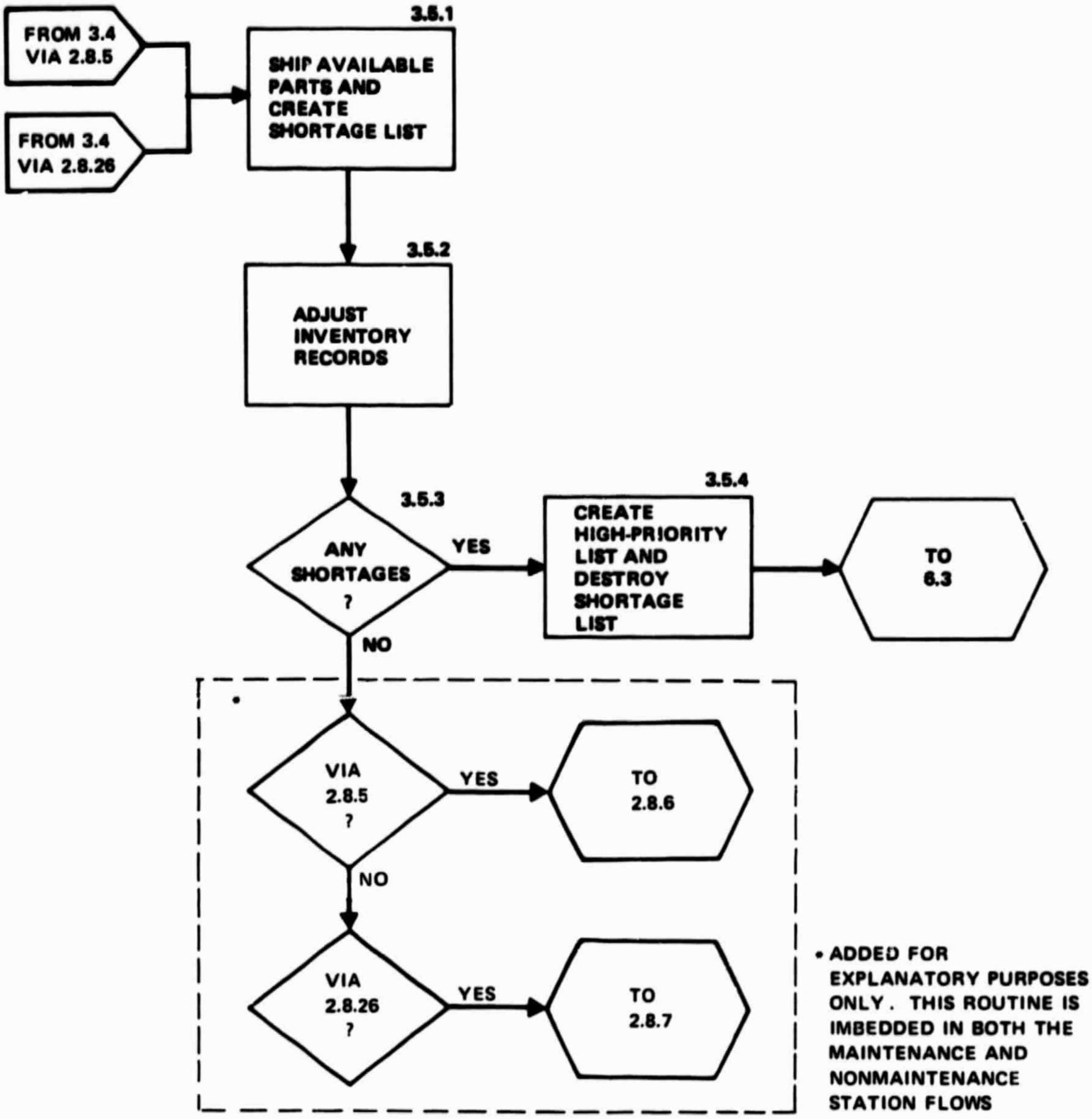


Figure 31. Process Emergency Parts Request (Section 3.5)

until the emergency request is satisfied. This has the effect of minimizing transit time for the parts that require ATE; note that ATE is the only potential bottleneck in the airline repair shop that is included in the simulation.

3.5.1 Ship Available Parts and Create Shortage List

Input

- Emergency parts request queue: from 3.4
- Emergency resupply time from parts depot: from line station data base, item 8
- Time tag on emergency request: from 2.8.5 or 2.8.26

Outputs

- Updated "time of last update" field in requesting station's inventory status file and in spares pool inventory status file
- Component part number and quantity shipped: to 3.5.2
- Shortage list: to 3.5.3, 3.5.4, 3.5.5

Processing

- REPLACE "TIME OF LAST UPDATE" FIELD IN REQUESTING STATION'S INVENTORY STATUS FILE BY EMERGENCY RESUPPLY TIME PLUS REQUEST TIME
- SIMILARLY REPLACE "TIME OF LAST UPDATE" FIELD IN PARTS DEPOT INVENTORY STATUS FILE BY TIME TAG ON EMERGENCY REQUEST
- WHILE COMPONENT PART NUMBERS REMAIN

```
        QUANTITY SHIPPED = MINIMUM OF QUANTITY REQUESTED AND
        QUANTITY ON HAND.
        IF QUANTITY SHIPPED LT QUANTITY REQUESTED,
        THEN DECREMENT QUANTITY FIELD IN REQUEST RECORD BY
        QUANTITY SHIPPED. CREATE SHORTAGE RECORD WITH COM-
        PONENT PART NUMBER AND QUANTITY REQUESTED
        ELSE DESTROY REQUEST RECORD
        ENDIF
    ENDWHILE
```

3.5.2 Adjust Inventory Records

Input

- Requesting station's inventory status file
- Parts depot inventory status file
- Component part number and quantity shipped: from 3.5.1

Output

- Incremented requesting station inventory
- Decremented parts depot inventory
- Shortage list: to 3.5.4

Processing

WHILE COMPONENT PART NUMBERS REMAIN

- DECREMENT PARTS DEPOT QUANTITY ON HAND BY QUANTITY SHIPPED AT TIME OF SHIPMENT

- INCREMENT REQUESTING STATION INVENTORY BY QUANTITY SHIPPED AT TIME RECEIVED

LOOP

3.5.3 Any Shortages?

Input

- Shortage list: from 3.5.1

Output

- YES: directs flow to 3.5.4
- NO: to END

Processing

IF SHORTAGE LIST VACANT
THEN NO
ELSE YES
ENDIF

3.5.4 Create High-Priority List and Destroy Shortage List

Input

- Shortage list: from 3.5.1
- Automatic test equipment (ATE) flag: from component data base, item 11
- Time of request: passed via 3.4
- Shortage tallies for components of interest at station of interest

Output

- High priority list for those out of stock components that require ATE in airline repair shop: to 6.3

Processing

```
WHILE COMPONENT PART NUMBERS REMAIN
     IF ATE FLAG SET
     THEN MOVE SHORTAGE RECORD FROM SHORTAGE LIST TO
     HIGH-PRIORITY LIST AND INCREMENT SHORTAGE TALLY FOR
     COMPONENT PART NUMBER BY QUANTITY SHORT
     ENDIF
ENDWHILE
     DESTROY SHORTAGE LIST
     TAG HIGH-PRIORITY LIST WITH TIME OF EMERGENCY REQUEST
```

3.6 Add Parts Request to Normal Queue

Input

- Time of request: from 2.8.9 or 2.8.24 or 4.2
- Requesting station
- Request list by component part number: from 2.8.9 or 2.8.24 or 4.2

Output

- Incremented normal FIFO queue
- Time of insertion: to 3.7

Processing

- APPEND NORMAL REQUEST LIST TO FIFO FORMAL QUEUE
- SET INSERTION TIME = REQUEST TIME

3.7 Process Routine Parts Requests

Input

- Normal shipping time from parts depot to line station: from line station data base, item 1
- Time tag on normal request: from 2.8.9, 2.8.24, or 4.2 via 3.6
- Inventory status table from requesting station
- Inventory status table from parts depot
- Normal parts request queue

Output

- Updated request queue
- Decremented parts depot inventory
- Incremented requesting station inventory

Processing

- REPLACE "TIME OF LAST UPDATE" FIELD IN REQUESTING STATIONS INVENTORY STATUS TABLE BY INSERTION TIME PLUS NORMAL SHIPPING TIME FROM PARTS DEPOT

- REPLACE "TIME OF LAST UPDATE" FIELD BY TIME OF SHIPMENT IN PARTS DEPOT INVENTORY STATUS TABLE
  WHILE COMPONENT PART NUMBERS REMAIN

  - DECREMENT PARTS DEPOT QUANTITY ON HAND BY QUANTITY SHIPPED AT TIME OF SHIPMENT

  - INCREMENT REQUESTING STATION INVENTORY BY QUANTITY SHIPPED AT TIME RECEIVED

  ENDWHILE
  END

5.4 SCHEDULED MAINTENANCE SIMULATION

Airlines practice considerable ingenuity in scheduling their airplanes to arrive at the proper time and place to receive scheduled maintenance. The CEM simulates this by tallying the parts used, accumulating the labor needed, and by performing scheduled FCS maintenance in zero time.

The logic for scheduled maintenance flow, shown in Figure 32, follows:

4.1 Restore FCS to Zero Failure State

Input

- Component status table (sec. 7.2.2)
- LRU inventory status table at hangar
- For each failed LRU part number

  - Repair time mean (first and additional): from component data base, items 9.2 and 9.3

- Operating time monitoring code: from component status table, item 5.1
- Arrival time at line station: from 1.8
- Accumulation of scheduled maintenance labor by LRU part number

Output

- Adjusted hangar inventory status table
- Updated component status table
- Parts request list: to 4.2
- Arrival time: to 4.2

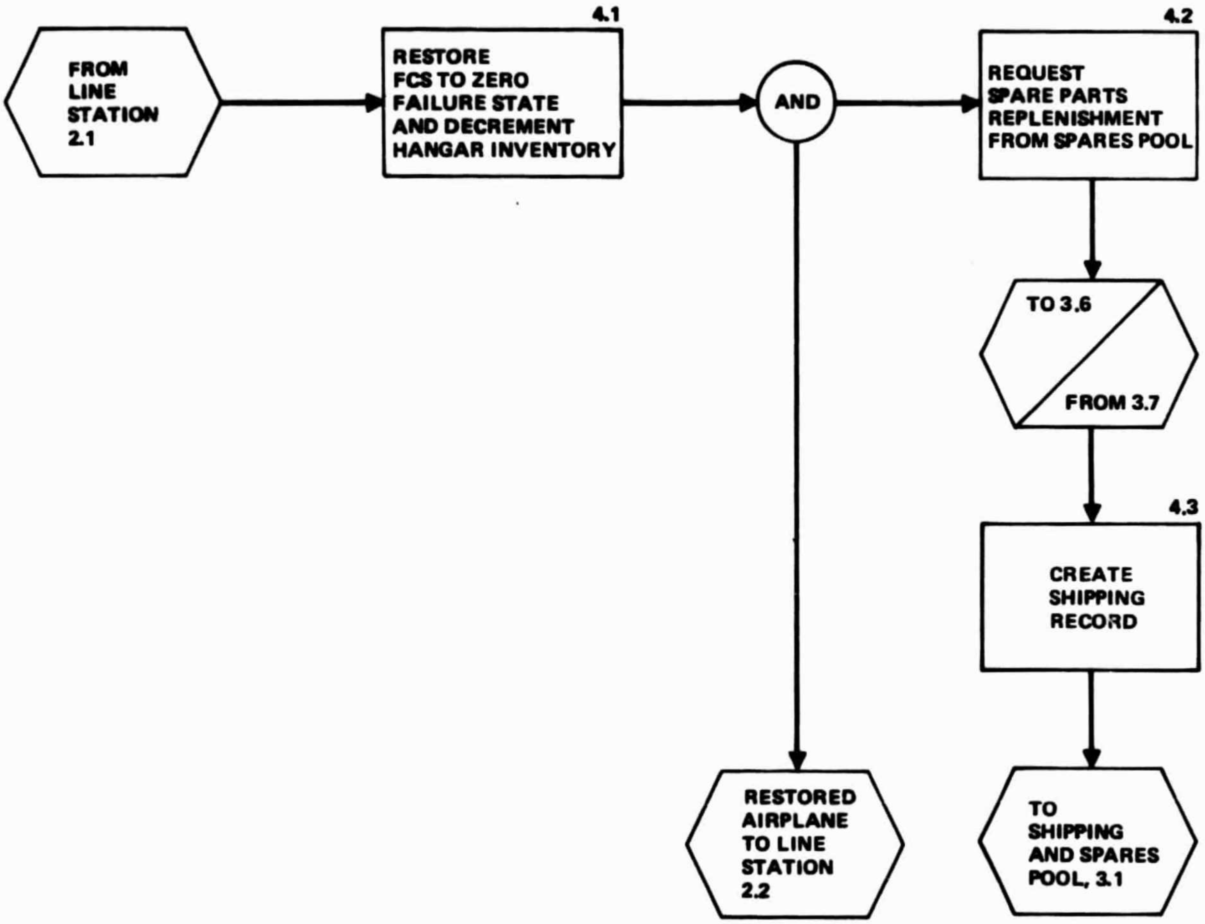


Figure 32. Scheduled Maintenance (Section 4.0)

Processing

- RESTORE ALL TYPE 2 LRUS CONTAINING FAILED SRUS AT THE TYPE 2 LEVEL

- RESTORE ALL TYPE 2 LRUS NOT CONTAINING FAILED SRUS AT THE TYPE 3 LEVEL

- NO PARTS OUTAGES WILL BE PERMITTED IN THE HANGAR; STOCKING LEVELS MUST BE ADEQUATE FOR THE DEMAND AT ALL TIMES

NOTE: When more than one maintenance task must be performed in the same work area, the time to perform the first task will frequently be longer than times for subsequent (additional) tasks. It will be assumed that half the failed LRUs of a given part number are "firsts" and half are "additionals." For greater accuracy, complex addressing considerations would have to be invoked.

LET Q(I) = QUANTITY OF FAILED LRUS HAVING I'TH PART NUMBER
N(I) = NUMBER OF MECHANICS REQUIRED TO RESTORE I'TH LRU
T1(I) = MEAN TIME TO RESTORE FIRST PIECE
T2(I) = MEAN TIME TO RESTORE ADDITIONAL PIECE
M(I) = LABOR HOURS TO RESTORE THE I'TH LRU

THEN, THE LABOR TO RESTORE THE I'TH LRU, M(I), IS GIVEN BY

M(I) = Q(I)*N(I) *(T1(I)+T2(I))/2
INCREMENT SCHEDULED MAINTENANCE LABOR HOURS FOR BOTH LRU AND FCS BY M(I)

THE LABOR HOURS TO RESTORE THE ENTIRE FCS IS GIVEN BY SUMMING THE M(I)S

- TALLY NUMBER OF FCS RESTORATIONS, OVERALL AND BY COMPONENT PART NUMBER QUANTITIES USED

- DECREMENT HANGAR INVENTORY STATUS TABLE BY Q(I) AND CREATE REQUEST LIST RECORD WITH LRU PART NUMBER AND QUANTITY USED

- DO NOT ADVANCE THE CLOCK

- SET AIRPLANE'S ACCUMULATED BLOCK TIME SINCE LAST SCHEDULED FCS MAINTENANCE TO ZERO

4.2 Request Spare Parts Replenishment From Spares Pool

Input

- Parts request list: from 4.1
- Arrival time of airplane: from 4.1

Output

- Parts request list: to 3.6

Processing

- PASS PARTS REQUEST LIST TO 3.6

  NOTE: EACH RECORD INCLUDES

  - LRU PART NUMBER
  - QUANTITY REQUIRED

- TIME-TAG THE REQUEST WITH SUBJECT AIRPLANE'S ARRIVAL TIME AT LINE STATION

4.3 Create Shipping Record

Input

- Repair code: from component data base, item 4.1

Output

- Shipping record

Processing

- IF REPAIR CODE = "VV"
- THEN SET SHIPPING CODE = 8
- ELSE SET SHIPPING CODE = 7
- ENDIF

  NOTE: Repair codes are:

  AA: airline test and airline repair
  AV: airline test and vendor repair
  AT: airline test and throwaway
  VV: vendor test and vendor repair

- CREATE SHIPPING RECORD WITH THESE FIELDS:

  - COMPONENT PART NUMBER
  - TIME AT COMPLETION OF VENDOR REPAIR
  - SHIPPING CODE
  - TIME OF SHIPMENT
  - "REPAIRED" FLAG SETTING

5.5 VENDOR REPAIR SIMULATION

The logic for vendor repair, shown in Figure 33, follows:

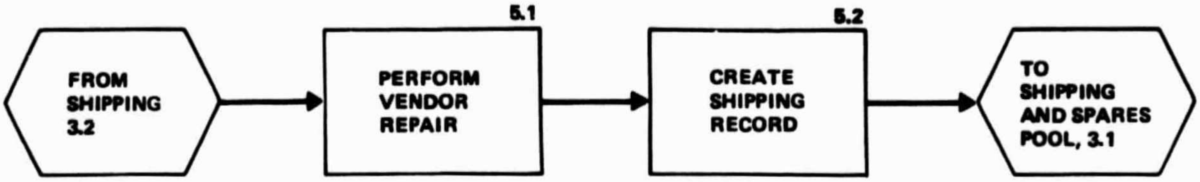


Figure 33. Vendor Repair (Section 5.0)

5.1 Perform Vendor Repair

Input

- Component part number and quantity: passed from 3.2
- Component repair code: passed from 3.2
- Operating time monitoring code: from component data base, item 8.1
- Vendor processing time: from component data base, item 4.2
- Arrival time: from 3.2
- Vendor repair tally by component part number

Output

- Time of repair completion: to 5.2
- Vendor repair flag setting: to 3.1
- Incremented vendor repair tally by component part number

Processing

- ADVANCE CLOCK BY VENDOR REPAIR TIME

- SET "REPAIRED" FLAG

- IF TIME MONITORED COMPONENT
  THEN RESET ELAPSED TIME TO ZERO
  ENDIF

- INCREMENT VENDOR REPAIR TALLY BY QUANTITY REPAIRED

5.2 Create Shipping Record

Input

- Repair code: from component data base, item 4.1
- Time of shipment from line station: from 2.8.10
- Time of repair completion: from 5.1

Output

- Shipping record

Processing

- SET "REPAIRED" FLAG

- IF REPAIR CODE = "VV"
  THEN SET SHIPPING CODE = 6
  ELSE SET SHIPPING CODE = 5

- ENDIF

- CREATE SHIPPING RECORD WITH THESE FIELDS:

  - COMPONENT PART NUMBER
  - TIME AT COMPLETION OF VENDOR REPAIR
  - SHIPPING CODE
  - TIME OF SHIPMENT
  - "REPAIRED" FLAG SETTING

5.6 AIRLINE REPAIR SHOP SIMULATION

At the time a type 1 or type 2 LRU enters the repair shop, a repair status record is created that contains the following fields, some of which are initially blank:

- Component part number
- Arrival time
- Departure time
- Test priority (default value = 1)
- ATE flag
- Position in ATE queue
- Start time if test underway
- Time to go of ATE or manual test
- Overtime limit (default value = 0)
- "Repaired" flag setting
- Hierarchy code

The only potential bottleneck in the repair shop simulation is ATE queueing. The logic for the airplane repair shop simulation, shown in Figure 34, follows:

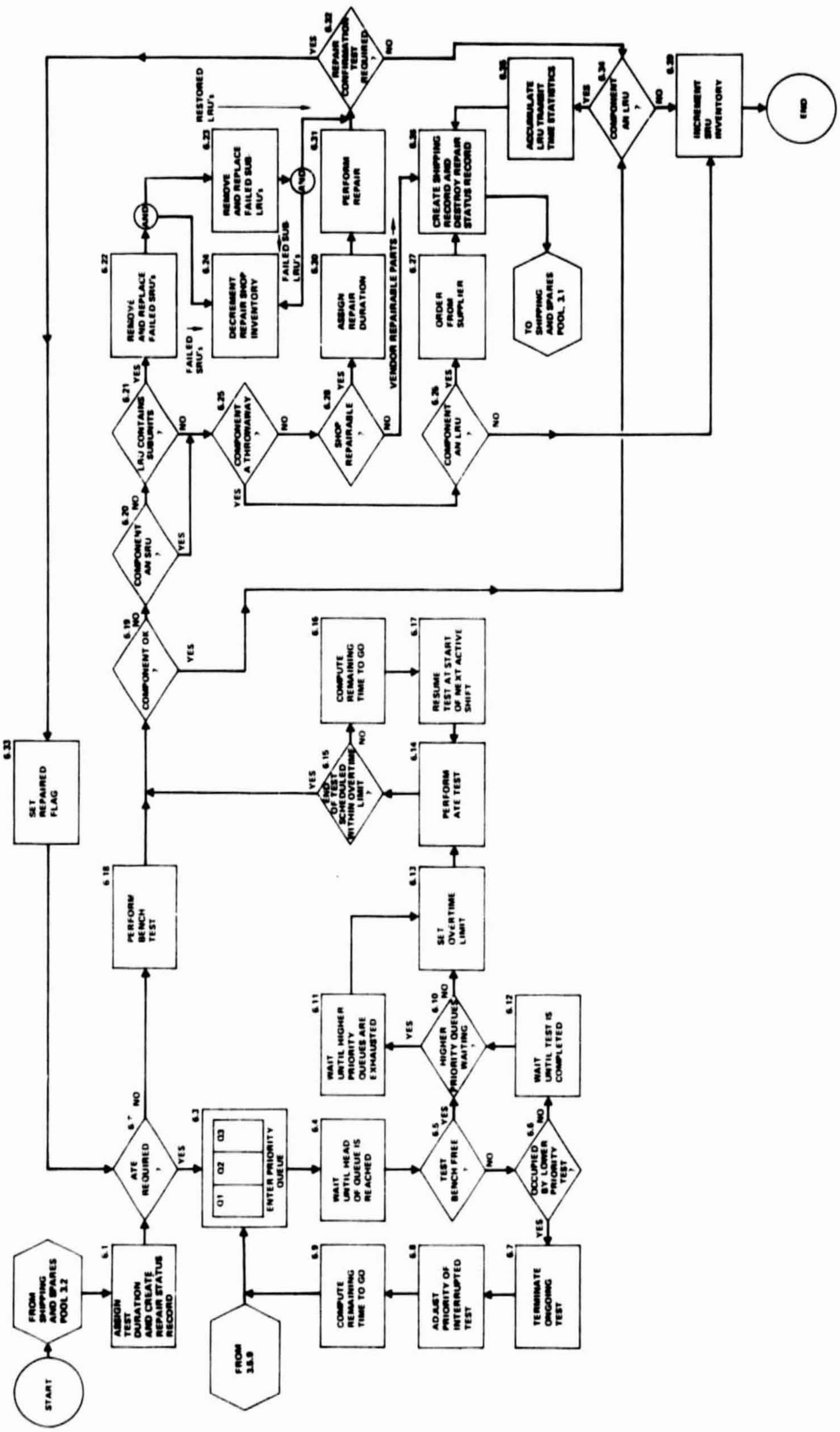


Figure 34. Airline Repair Shop (Section 6.0)

6.1 Assign Test Duration and Create Repair Status Record

Input

- Repair shop test time probability density code and parameters: from component data base, items 13.1 and 13.2

- Time now: from 3.2

Output

- Time-to-go at start of test

  - This is used to advance the clock for ATE and manual test times (secs. 6.14 and 6.18)

- Entry time of the component: to 6.35 and 6.36

Processing

- CREATE REPAIR STATUS RECORD AS SPECIFIED IN 7.2.3

- PERFORM DRAW FROM APPROPRIATE DENSITY FUNCTION AND INITIALIZE TEST TIME-TO-GO

- IF COMPONENT IS AN LRU, TAG IT WITH ITS ENTRY TIME
  ENDIF

6.2 ATE Required?

Input

- Airline repair shop test code for each component of interest to denote automatic test equipment (ATE) or manual test equipment (MTE): from component data base, item 11

Output

- YES: directs flow to 6.3
- NO: to 6.18

Processing

- IF AIRLINE REPAIR SHOP TEST CODE = "A"
- THEN YES
- ELSE NO
- ENDIF

6.3 Enter Priority Queue

Input

- Priority of the repair action: from repair status record
- Shortage notice: from 3.5.1

Output

- Time of entry of the component
- Initial queue position in priority 1, 2, or 3 queue

Processing

IF NO SHORTAGE RECORDS CORRESPONDING TO COMPONENT PART NUMBER
THEN PLACE PART IN PRIORITY QUEUE DESIGNATED BY REPAIR STATUS RECORD
ELSE
    IF DESIRED PARTS ARE WAITING IN P(1) OR P(2) QUEUES
    THEN INCREASE PRIORITY TO P(3) AND MOVE PART INTO P(3) FIFO QUEUE, REPEATING THIS UNTIL REQUIRED QUANTITY IS REPRIORITIZED OR UNTIL PARTS ON HAND ARE EXHAUSTED
    ELSE
    WAIT UNTIL DESIRED PART ARRIVES IN P(1) OR P(2) QUEUE AND REPRIORITIZE AS ABOVE UNTIL DESIRED QUANTITY IS REPRIORITIZED
    ENDIF
ENDIF

6.4 Wait Until Head of Queue Is Reached

Input

- Queue position, triggered by 6.3

Output

- Queue position

Processing

- AS EACH COMPONENT EXITS QUEUE, SUCCEEDING ONES ARE DECREMENTED IN POSITION; HANDLED VIA SIMULATION UTILITIES

6.5 Test Bench Free?

Input

- Test bench status, occupied or free, from simulation utilities

Output

- YES: directs flow to 6.10
- NO: to 6.6

Processing

- IF TEST BENCH FREE
- THEN YES
- ELSE NO
- ENDIF

6.6 Occupied by Lower Priority Test?

Input

- Priority of test in progress
- Priority of the test at head of queue

Output

- YES: directs flow to 6.7
- NO: to 6.12

Processing

- IF PRIORITY OF TEST IN PROGRESS LT PRIORITY OF TEST AT HEAD OF QUEUE
- THEN YES
- ELSE NO
- ENDIF

6.7 Terminate Ongoing Test

Input

- Time now: internally generated
- Start time of ongoing test: from repair status record of component being tested
- Test duration of ongoing test: from above source

Output

- Time-to-go of ongoing test; used in 6.9
- Priority of ongoing test; used in 6.8
- Stop time of ongoing test, used in 6.14

Processing

- TIME-TO-GO OF ONGOING TEST AT STOP TIME
  = TIME-TO-GO AT START OF ONGOING TEST
  - (TIME NOW - START TIME OF ONGOING TEST)

6.8 Adjust Priority of Interrupted Test

Input

- Priority of interrupted test: from repair status record 7.2.3, item 5

Output

- New priority of interrupted test: to 6.3

Processing

- IF PRIORITY = P(1)
- THEN SET PRIORITY = P(2)
- ELSE CONTINUE
- ENDIF

6.9 Compute Remaining Time-To-Go (of Interrupted Test)

Input

- Time-to-go at stop time: from 6.7

Output

- Remaining time-to-go of interrupted test: to 6.3

Processing

- REMAINING TIME-TO-GO =
  TIME-TO-GO OF ONGOING TEST AT STOP TIME
  + 15 MINUTES

6.10 Are Higher Priority Queues Waiting?

Input

- Priority of occupied queues other than that of subject test
- Priority of subject test

Output

- YES or NO (flow directed to 6.11 or 6.13 respectively)

Processing

- COMPARE PRIORITY OF ALL COMPONENTS AT HEAD OF QUEUES
- IF ITEMS OF HIGHER PRIORITY EXIST
- THEN YES
- ELSE NO
- ENDIF

6.11 Wait Until Higher Priority Queues Are Exhausted

Input

- Priority of test: from repair status record, item 5

- Occupancy of higher priority queues (internally generated by SIMSCRIPT utilities)

Output

- Event notice that higher priority queues are vacant and the time; used in 6.14

Processing

- VIA SIMULATION UTILITIES

6.12 Wait Until Test Is Completed

Input

- Time-to-go of ongoing test; block triggered by "NO" from 6.6

Output

- Event notice that ATE is free, directs flow to 6.10

Processing

- QUEUE UNTIL COMPLETION OF ONGOING TEST

6.13 Set Overtime Limit

Input

- Test priority, passed through by 6.3
- Overtime limit for priority (2) tests from repair shop data base, item 3

Output

- Overtime limit for the test, hours; used in 6.15

Processing

```
IF PRIORITY = P(3), OVERTIME LIMIT = INFINITY
ELSE,
    IF PRIORITY = P(2), OVERTIME LIMIT IS FIXED
ELSE,
    IF PRIORITY = P(1), OVERTIME = 0
ENDIF
```

NOTE: When overtime extends into a working shift, it continues and the mechanic performing the test continues until completion.

Overtime is determined on an individual part basis and enables continuation of a test past the normal end of shift. However, only priority 3 tests may begin after shift end. Lower priority tests, regardless of their overtime limits, are not to begin outside of normal shift times.

6.14 Perform ATE Test

NOTE: If priority = P3, test begins immediately. Otherwise test cannot begin until normal shift time exists.

Input

1. Time-to-go at start of test: from 6.1, 6.9, or 6.16
2. Starting time of test: internally generated
3. Active shift start times and end times from repair shop data base, items 1.2 and 1.3
4. Overtime limit: from 6.13
5. Previous regular time total for ATE: individual and total
6. Previous overtime total for ATE: taken over all servers
7. Previous regular time total for affected component part number
8. Previous overtime total for affected component part number

Output

- Event notice with
    - Time-to-go at start
    - Start time
- Updated totals for items 5 through 8 in input list

Processing

- UNLESS INTERRUPTED BY HIGHER PRIORITY TEST PERFORM TEST UNTIL COMPLETED WITHIN SHIFT TIME OR TERMINATED BY EXHAUSTION OF OVERTIME
- INCREMENT ITEMS 5 THROUGH 8 IN INPUT LIST

6.15 Can Test Be Completed Within Overtime Limit?

Input

- Active shift end times: passed through 6.14
- Start time of test: from 6.14
- Time-to-go at start of test: passed through 6.3
- Overtime limit: from 6.13

Output

- YES: directs flow to 6.19
- NO: directs flow to 6.16
- Stop time of test

Processing

- STOP TIME OF TEST = ENDING TIME OF LAST SHIFT + OVERTIME LIMIT
- IF (STARTING TIME + TIME-TO-GO) LE STOP TIME OF TEST
- THEN YES
- ELSE NO
- ENDIF

6.16 Compute Remaining Time-To-Go

Input

- Time-to-go at start of test: via 6.3
- Start time of test: from 6.14
- Stop time of test: from 6.15

Output

- Remaining time-to-go of test: to 6.14

Processing

REMAINING TIME-TO-GO =
    TIME-TO-GO AT START OF TEST - (STOP TIME - START TIME OF TEST)

6.17 Resume Test at Start of Next Active Shift

Input

- Start time of next shift: from repair shop data base

Output

- Start time of next shift: to 6.14

Processing

- ADVANCE SIMULATION CLOCK TO START TIME OF NEXT SHIFT

6.18 Perform Bench Test

Input

- Assigned test duration: from 6.1

Output

- Accumulation of test time by component part number and total: to final output

Processing

- ACCUMULATE BENCH TEST TIME BY COMPONENT PART NUMBER AND OVERALL

- ADVANCE SIMULATION CLOCK BY DURATION OF TEST

  NOTE: This accumulation is initialized to zero at the beginning of simulation run and is incremented throughout the run. It is assumed that manual test personnel and equipment are available on demand without queuing.

6.19 Check OK?

This block enables a repair code "AV" part (airline test and vendor repair) to reenter the airline repair shop's test sequence to verify the adequacy of the vendor's repair. Such parts will have a shipping code 5 assigned in the vendor repair simulation; otherwise, repair code 6 will be assigned, causing the part to be shipped directly to the parts depot.

Input

- Probability of unconfirmed failure of the component: from component data base, item 6

- "Repaired" flag: from 6.33

Output

- YES: directs flow to 6.34
- NO: directs flow to 6.20

Processing

- IF REPAIR FLAG SET

- THEN YES

- ELSE DRAW FROM UNIFORM DISTRIBUTION (0,1)
  IF VALUE GE PROBABILITY OF UNCONFIRMED FAILURE
  THEN YES
  ELSE NO
  ENDIF
  ENDIF

6.20 Component an SRU?

Input

- Component hierarchy code: from repair status record, item 2

Output

- YES: directs flow to 6.25
- NO: directs flow to 6.21

Processing

- IF COMPONENT HIERARCHY CODE = 4 (sec. 7.1.1, item 3)
- THEN YES
- ELSE NO
- ENDIF

6.21 LRU Contain Subunits?

Input

- Component hierarchy code: from repair status record, item 2

Output

- YES: directs from to 6.22
- NO: directs flow to 6.25

Processing

- IF HIERARCHY CODE = 2
- THEN YES

- ELSE NO
- ENDIF

6.22 Remove and Replace Failed SRUs

Input

1. Identification of failed SRUs by part number and address: from 1.6.6
2. Repair shop remove and replace time for first and additional units of this SRU from component data base, items 9.2 and 9.3
3. Previous number of failed units for affected SRU part number
4. Previous repair time total for the SRU part number
5. Previous total repair time for this shop
6. Previous total repair time for this airplane model
7. Previous total repair time for subject FCS

Output

- SRU quantities replaced by part number: to 6.24
- Repair status records for removed SRU
- Items 3 through 7 in input list are updated and held until the next activation of 6.22. These sums appear in final output.

Processing

- DETERMINE FIRST AND ADDITIONAL QUANTITIES THUS:
  WHILE SRU PART NUMBERS REMAIN IN THIS TYPE 2 LRU
    FIRST QUANTITY = 1
    ADDITIONAL QUANTITY = QUANTITY OF ADDRESSES CONTAINING FAILED SRUS OF THIS PART NUMBER - 1
  TOTAL QUANTITY REPLACED = QUANTITY OF ADDRESSES HAVING FAILED UNITS

- COMPUTE REMOVE AND REPLACE TIMES AS FOLLOWS:
    REMOVE AND REPLACE TIME FOR AFFECTED SRUS = (QUANTITY OF FIRST ITEMS)* (TIME FOR FIRST ITEM) + (QUANTITY OF ADDITIONAL ITEMS)* (TIME FOR EACH ADDITIONAL ITEM)

- INCREMENT ITEMS 3 THROUGH 7 ON INPUT LIST

- CREATE REPAIR STATUS RECORDS FOR REMOVED SRUS AS FOLLOWS: (SEE 7.2.3 FOR FIELD LIST)

- SET HIERARCHY CODE = 4

NOTE: The arrival time field for the SRU will be set equal to the arrival time of the LRU from which it was removed. Items 6 and 7 in repair status record do not apply.

6.23 Remove and Replace Failed Sub-LRUs

NOTE: A type 3 LRU (sub-LRU) is an LRU contained within a type 2 LRU.

Input

1. Identification of failed LRUs by part number and address: from 1.6.6
2. Repair shop remove and replace time for first and additional units of this LRU: from component data base, items 9.2 and 9.3
3. Previous number of repair actions for the LRU part number
4. Previous repair time total for the LRU
5. Previous total repair time for this shop
6. Previous overall repair time for this airplane model
7. Previous overall repair time for subject FCS
8. ATE code: from component status record, item 11

Output

- Sub-LRU quantities replaced by part number: to 6.24
- Repair status records for removed sub-LRUs
- Items 3 through 7 on input list are updated and held until the next activation of this block. These sums appear in final output.

Processing

- DETERMINE FIRST AND ADDITIONAL QUANTITIES THUS:
  WHILE SUB-LRU PART NUMBERS REMAIN IN THIS TYPE 2 LRU
  FIRST QUANTITY = 1
  ADDITIONAL QUANTITY = QUANTITY OF ADDRESSES CONTAINING FAILED SRU OF THIS PART NUMBER - 1
  TOTAL QUANTITY REPLACED = QUANTITY OF ADDRESSES HAVING A FAILED UNIT

- COMPUTE REMOVE AND REPLACE TIMES AS FOLLOWS:
  REMOVE AND REPLACE TIME FOR AFFECTED SRUS = (QUANTITY OF FIRST ITEMS)* (TIME FOR FIRST ITEM) + (QUANTITY OF ADDITIONAL ITEMS)* (TIME FOR EACH ADDITIONAL ITEM)

- INCREMENT ITEMS 3 THROUGH 7 ON INPUT LIST

- CREATE REPAIR STATUS RECORDS FOR REMOVED LRUS AS FOLLOWS: (SEE 7.2.3)

  - SET HIERARCHY CODE = 3

  NOTE: The arrival time field will be equal to the arrival time, item 3, of the LRU from which it was removed.

  - IF REPAIR SHOP TEST CODE = "A," THEN SET ATE FLAG

6.24 Decrement Repair Shop Inventory

Input

- Stock level of the subject component; the initial level is obtained from the repair shop's inventory status table

- Component quantities replaced by component ID number: from 6.23 or 6.24

Output

- Updated inventory status record for the subject components

Processing

- DECREMENT STOCK LEVEL OF AFFECTED COMPONENTS

6.25 Component a Throwaway?

Input

- Component repair code: from component data base, item 4.1

Output

- YES: directs flow to 6.26
- NO: directs flow to 6.28

Processing

- IF COMPONENT REPAIR CODE = AT
- THEN YES
- ELSE NO
- ENDIF

6.26 Is Component an LRU?

Input

- Component hierarchy code: from repair status record, item 2

Output

- YES: directs flow to 6.27
- NO: to 6.29

Processing

- IF HIERARCHY CODE = 4
- THEN NO
- ELSE YES
- ENDIF

6.27 Order From Supplier

Input

- Reorder time delay for throwaway LRUs: from component data base, item 4.3

Output

- Reorder time delay for affected LRU, after which spares pool is incremented: to 3.1

Processing

- INCREMENT SPARES POOL BY 1 FOR EACH AFFECTED PART NUMBER AFTER THE SPECIFIED REORDER TIME DELAY

  NOTE: The shipping and spares pool module maintains inventory control of LRUs at line stations, hangar, and the repair shop. Inventory control is not required for SRUs in this model.

6.28 Shop Repairable?

Input

- Component repair code: from component data base, item 4.1

Output

- YES: directs flow to 6.30
  NO: directs flow to 6.36

- Vendor repair flag setting

Processing

- IF COMPONENT REPAIR CODE = "AA"
  THEN YES
  ELSE NO

- SET VENDOR REPAIR FLAG
- ENDIF

6.29 Increment SRU Inventory

Input

- Activated by "NO" from either 6.26 or 6.29
- Current inventory level of affected component

Output

- Updated SRU inventory level (ultimately this will be used to compute the size of the SRU pool and the investments required).

Processing

- INCREMENT INVENTORY OF AFFECTED SRU BY 1 FOR EACH ONE PROCESSED; DO THIS WITH ZERO TIME DELAY

  NOTE: Although the CEM does not manage SRU inventory levels, for economic analysis purposes it is necessary to know how many SRUs are in process in the repair shop.

6.30 Assign Repair Duration

Input

- Shop repair time probability density type and parameters: from component data base, items 9.1 and 9.2

Output

- Repair time duration: to 6.31

Processing

- PERFORM DRAW FROM SPECIFIED DENSITY FUNCTION FOR THE COMPONENT

6.31 Perform Repair

Input

1. Repair duration: from 6.30
2. Previous total repair time for the affected component ID number
3. Previous total repair time for FCS
4. Previous total repair time for airplane

Output

- Updated total repair time for the affected component: used in final CEM output and for economic analysis

Processing

- INCREMENT TOTAL REPAIR TIME BY REPAIR DURATION FOR THE AFFECTED COMPONENT
- IF COMPONENT IS TIME MONITORED, SET ACCUMULATED OPERATING TIME TO ZERO
- ENDIF

6.32 Repair Confirmation Test Required?

Input

- Repair confirmation code: from component data base, item 12

Output

- YES: directs flow to 6.33
- NO: to 6.34

Processing

- IF REPAIR CONFIRMATION CODE = "Y" FOR AFFECTED COMPONENT
- THEN YES
- ELSE NO
- ENDIF

6.33 Set "Repaired" Flag

Input

- Component status record

Output

- Repair code set for affected component

Processing

- SET REPAIR FLAG

6.34 Component an LRU?

Input

- Component hierarchy code: from component data base, item 3

Output

- YES: directs flow to 6.35
- NO: directs flow to 6.29

Processing

- IF HIERARCHY CODE = 1 OR 2 OR 3

- THEN YES

- ELSE IF HIERARCHY CODE = 4
  THEN NO
  ENDIF

- ENDIF

6.35 Accumulate LRU Transit Time Statistics

Input

- Hierarchy code: from FCS component status table, item 6

- Entry time: from 6.1

- Exit time: internally generated by accumulating all processing and waiting times

- Tally of repair actions for type 1 and type 2 LRUs by part number

- Accumulated transit time for type 1 and type 2 LRUs by part number

Output

- Updated transit time accumulation for type 1 and type 2 LRUs by part number

Processing

- IF HIERARCHY CODE = 1 OR 2

- THEN INCREMENT REPAIR TALLY BY 1 FOR COMPONENT PART NUMBER OF INTEREST AND INCREMENT TRANSIT TIME ACCUMULATION BY (EXIT TIME MINUS ENTRY TIME)
  ELSE CONTINUE
  ENDIF

6.36 Create Shipping Record and Destroy Repair Status Record

Input

- Component part number

- "Repaired" flag setting: from 6.33 or 5.1

- Component repair code: from component data base, item 3

- Time at completion of final repair shop operation on component: internally generated in 6.14 or 6.18 or 6.31

- Repair status record for component of interest

Output

Shipping record

Processing

NOTE: See the beginning of Section 5.3, SHIPPING AND INVENTORY MANAGEMENT SIMULATION, for list of shipping codes.

- IF "REPAIRED" FLAG SET
  THEN SET SHIPPING CODE = 3
  ELSE SET SHIPPING CODE = 4 (THESE COMPONENTS WOULD HAVE AN "AV" SHIPPING CODE)
  ENDIF

  NOTE: Repair codes are:

  AA: airline test and airline repair
  AV: airline test and vendor repair
  AT: airline test and throwaway
  VV: vendor test and vendor repair

- CREATE SHIPPING RECORD WITH THESE FIELDS:

  - COMPONENT PART NUMBER
  - TIME AT COMPLETION OF FINAL REPAIR OR TEST OPERATION PERFORMED ON PART IN AIRLINE REPAIR SHOP
  - SHIPPING CODE
  - QUANTITY SHIPPED

- DESTROY REPAIR STATUS RECORD

5.7 DATA BASES

Information required to drive the CEM can be conveniently categorized into two classes: input data bases and processing data bases. The input data is static and the processing data is dynamic.

It is noted that these processing bases are envisioned as convenient, temporary data entities that minimize the number of accesses to the input data bases. The implementor is free to reorganize this structure as he sees fit within the general constraints of modularity.

7.1 Input Data Bases

7.1.1 Component Data Base

Item 1 Component name

Item 2 Component ID number

Item 3 Hierarchy code

1 = LRU that has no subunits and is not contained in any other LRU
2 = an LRU that contains subunits
3 = an LRU that is a subunit of a type 2 LRU
4 = an SRU that is a subunit of a type 2 LRU

Item 4.1 Repair code

AA = airline test and airline repair
AV = airline test, vendor repair, airline retest
AT = airline test and throwaway
VV = vendor test and vendor repair

Item 4.2 Vendor processing time (for repair codes AV and VV)

Item 4.3 Reorder time for throwaway components

Item 5 Attrition rate (quantity of parts scrapped per repair action)

Item 6 Probability of unconfirmed failure

Item 7 Failure visibility proportion, postflight
(preflight proportion = 1 - postflight proportion)

Item 8.1 Operating time monitoring code

T = time monitored
U = unmonitored

Item 8.2 Operating time limit, hours
This and the next item are required only for time-monitored FCS components (code T)

Item 8.3 Failure rate table (operating hours versus failures per hour)

Item 8.4 Constant failure rate
This is required for components that are not time-monitored (code U)

Item 9.1 Line maintenance remove and replace time probability density code

U = uniform distribution (minimum, maximum)
G = truncated Gaussian (mean, standard deviation, maximum value)

F	=	fixed, nonstochastic value (mean)
T	=	triangular (minimum, mode, maximum)
L	=	truncated lognormal (mean, standard deviation, maximum value)

Item 9.2 Parameter value list for first component (up to three items)

Item 9.3 Parameter value list for additional component (up to three times)

Item 10 Number of mechanics required per line maintenance action

Item 11 Airline repair shop test code

A	=	ATE (automatic test equipment)
M	=	MTE (manual test equipment)

Item 12 Repair confirmation code

Y	=	repair confirmation required
N	=	repair confirmation not required

Item 13.1 Repair shop ATE test time probability density code (Item 9.1 contains list of codes)

Item 13.2 Parameter value list (up to three parameters)

Item 14.1 Component repair time probability density code (Item 9.1)

Item 14.2 Parameter value list (up to three parameters)

7.1.2 Line Station Data Base

Global values for all stations:

Item 1 Irrevocable time limit for airplane flight assignments

Item 2 Default file of minimum dispatch quantities

Each record in this file contains two fields:

- Stage ID number
- Minimum dispatch quantity of unfailed units

Item 3 Most demanding fleet dispatch requirements file

Each record contains two fields for each stage comprising the FCS

- Stage ID number
- Most demanding dispatch quantity of unfailed units

Item 4.1 Time-to-go within which maintenance tasks will not be interrupted

Item 4.2 Percentage of job completion after which maintenance tasks will not be interrupted

Item 5 Intertask transit time

Item 6 Average time for relocating subject airplanes in event of cancellation

Item 7 Increase in time-to-go for interrupted tasks

Item 8 Emergency LRU resupply time from parts depot

Item 9 Cutoff time after which cancelled airplanes are given high priority

Item 10 Standard hours worked per day

For Each Station:

Item 10 City name

Item 12 Station ID (LAX, ORD, etc.)

Item 13.1 Maintenance capability code

M = maintenance capability present
Z = no maintenance capability

Item 13.2 "Parent" station ID for nonmaintenance, code Z, stations

Item 13.3 Mechanic travel time from parent station

Item 13.4 Mechanic travel time to parent station

Item 14.1 Line station shift start and stop times, for maintenance, code M, stations

Item 14.2 Maintenance labor levels by shift (for code M stations)

Item 14.3 Manpower minimum depletion levels by shift (for code M stations)

Item 15 Initial inventory table

Each record has two fields:

- Component part number
- Initial quantity on hand

Item 16 Irrevocable time limit for flight assignment override (this overrides item 1)

Item 17 Intertask transit time (override of item 5)

Item 18 Most demanding station dispatch requirements file

Each record includes two fields:

- Stage ID number
- Most demanding dispatch quantity of unfailed units for the station

7.1.3 Airplane Data Base

For Each Airplane Model:

Item 1 Airplane model

Item 2.1 Subject airplane code

S = subject airplane
Q = other airplane

Item 2.2 Scheduled maintenance interval for subject FCS (blank for code "Q")

Item 2.3 LRU part number associated with nonsubject airplane (blank for code "S")

NOTE: All the line maintenance and failure characteristics of airplanes competing with the subject airplane are expressed in terms of a pseudo-LRU, which may have multiple failures per flight leg.

Item 3 Preflight checkout time

Item 4 Postflight checkout time

Item 5.1 Unscheduled removals per block hour for non-FCS systems aboard subject airplane, blank for code "Q" airplane

Item 5.2 Unscheduled removals per block hour for other airplanes, blank for code "S" airplane

Each Airplane:

Item 6 Tail number

7.1.4 Shipping and Inventory Data Base

There are 10 shipping actions of interest that are listed in the beginning of section 3.0. They are repeated here for ease of reference.

SHIPPING ACTION

SHIPPING CODE	FROM	TO
1	Line station	Repair shop
2	Line station	Vendor
3	Repair shop	Parts depot
4	Repair shop	Vendor
5	Vendor	Repair shop
6	Vendor	Parts depot
7	Hangar	Repair shop
8	Hangar	Vendor
9	Parts depot	Line station
10	Parts depot	Hangar

Shipping times to and from the vendor are an individual part characteristic and could logically be included in the component data base; similarly, shipping times associated with line stations could be included in the line station data base. However, in the interests of modularity, shipping information is specified here. The implementor is free to include shipping data in the component or line station data base.

Item 1 For all shipping codes not involving vendors, namely 1, 3, 7, 9, and 10, each record has three fields:

Shipping station ID, shipping code, shipping time.

Item 2 For all shipping codes involving vendors namely 2, 4, 5, 6 and 8, the data base is organized by part number. Each record contains the following fields:

Component part number, shipping code, shipping time. Note that the CEM assumes that shipping times to the vendor from any line station, parts depot, or hangar are independent of location.

Item 3 Stocking levels at start of simulation. The records in this file are comprised thus:

Station ID, component part number, quantity initially on hand.

7.1.5 FCS System Architecture Data Base

Capturing FCS characteristics in the CEM requires data in five major categories:

- Address affiliation
- Stage membership
- Dependent effects
- Availability criteria
- Dispatch minimums

A full discussion of the abstractions involved and an example is given in Section 2.8.2.

Item 1 Address affiliation file

This involves data organized thus:

1.1 Address
1.2 Component part number
1.3 Next higher LRU address
1.4 ID number of stage with which address is affiliated

Item 2 Stage membership file (derived from item 1)

The fields required are these for each stage

Stage ID number
List of addresses comprising the stage

Item 3 Dependent effects table

Causing address or causing stage
List of affected addresses with effect codes for each causing address

Effects codes are:

F = failed
A = nullified hot
D = nullified cold

Item 4 Availability of type 2 LRUs

LRU address
Stage ID numbers for stages contained wholly within the LRU part number
Minimum quantity of available addresses within above stages such that the type 2 LRU remains functional

Dispatch minimums (7.1.2.2) and flight schedule data base (7.1.2.8)

7.1.6 Airline Repair Shop Data Base

Item 1 Personnel loading

This is a table with the following data:

Shift start time, stop time, number of mechanics on duty

Item 2 Quantity of ATE stations
Item 3 Overtime limit for priority (2), interrupted tests

7.1.7 Flight Schedule Data Base

The flight schedule data base is a file sorted by schedule departure time. Each line station has its own file. The data in each flight record are these:

Item 1 Flight number
Item 2.1 Destination code
Item 2.2 Local time zone correction
Item 3 Airplane model
Item 4 Scheduled departure time
Item 5 Nominal block time
Item 6 Flight code

T = through flight
Z = originating flight
N = nonrevenue flight

Item 7 Delay time that results in a cancellation

Item 8 Lookup table: override of item 2 in line station data base—minimum dispatch quantities by stage ID number and quantity of unfailed units

7.2 Processing Data Bases

7.2.1 Inventory Status Tables

Each record in this data base includes the following fields:

- Station ID code
- Component part number
- Quantity on hand
- Time of last update

7.2.2 FCS Component Status Table

Each FCS, identified by airplane tail number, has a component status table with these fields for each LRU and SRU that comprises the FCS.

Item 1.1 Address
Item 1.2 Dependent event flag (set if failure causes dependent effects)
Item 1.3 "Cold" standby flag (if address is normally in "cold" standby)

Item 2 Component part number
Item 3 Status code

1	=	OK
2	=	failed
3	=	due for replacement, for time monitored components that have exceeded their replacement time limit
4	=	nullified "hot" (energized)
5	=	nullifed "cold" (unenergized)
6	=	in cold standby

Item 4 Failure visibility code

PRE	=	visible at first preflight check following failure
POST	=	visible at first postflight check following failure
KNOWN	=	always visible at first preflight check following failure

Item 5.1 Time Monitor Code

M	=	monitored
U	=	unmonitored
If code	=	M, then

Item 5.2 Operating time limit
Item 5.3 Accumulated operating time
Item 5.4 "Over Limit" flag setting, set if item 5.3 GT item 5.2
Item 5.5 Serial number
Item 6 Component hierarchy code

In addition, each type 2 LRU, when removed, must carry with it to the repair shop information as to the status of its type 3 LRUs and type 4 SRUs.

Item 7.1 Component part number
Item 7.2 Quantity failed

7.2.3 Airline Repair Status File

At the time a type 1 or type 2 LRU enters the repair shop, a repair status record is created containing the following fields, some of which are initially blank:

Item 1 Component part number
Item 2 Hierarchy code
Item 3 Arrival time
Item 4 Departure time
Item 5 Test priority (default value = P(1))
Item 6 ATE flag
Item 7 Position in ATE queue
Item 8 Time to go of ATE or manual test
Item 9 Overtime limit (default value = 0)
Item 10 "Repaired" flag setting
Item 11 Start time of test in progress

A component status record must also be created whenever failed type 3 LRUs and type 4 SRUs are removed from the type 2 LRU that contained them. This occurs in 6.22 and 6.23.

7.2.4 Line Station Mechanic Status Table

Each maintenance-capable line station, code M, will have a status table, the records of which include these fields as a minimum.

- Mechanic ID number
- Idle flag
- Task descriptors if not idle
    - Airplane tail number
    - Remote AOG flag (if set, task is uninterruptable)
    - Priority of task
- Normal shift assignment
- Current shift assignment

7.2.5 Airplane Status Record

Item 1.1 Airplane tail number

Item 1.2 Airplane model

Item 1.3 Subject airplane flag setting

Item 2 Virtual airplane flag setting

Item 3.1 Departing flight number to which assigned

Item 3.2 Destination code of assigned flight

Item 3.3 Through-flight flag setting

Item 3.4 Scheduled departure time

Item 3.5 Actual departure time

Item 3.6 Local time zone correction

Item 3.7 Actual arrival time

Item 4 Accumulated block time on subject airplane since last FCS scheduled maintenance

Item 5 Selection pool flag setting

Item 6 Disassigned flag setting

Item 7 "Existing" flag setting

Item 8 "Under Repair" flag setting

5.8 OUTPUTS OF THE COMPREHENSIVE EVALUATION MODEL

The following CEM outputs include those required to manually prepare ACES input to perform economic analyses of candidate design or to perform validity checking. The outputs are presented in the numerical order of the flow chart entities in which they are generated.

1. Preflight FCS system states (1.4).
   For each k-of-n stage in the subject FCS, the preflight occurrences of k = 0,1,2,...n will be tallied.

2. Postflight FCS system states (1.7).
   For each k-of-n stage in the subject FCS, the postflight occurrences of k = 0,1,2,...n will be tallied.

3. Total postflight checkout delay time at each line station (2.3).

4. Total number of arrivals by line station (2.3). (Note that this should be equal to the total number of departures.)

5. Through-flight substitution tally (2.7.2)

6. Total overtime hours worked at each maintenance line station (2.8.6).

7. Total line station labor-hours for mandatory unscheduled maintenance by LRU part number (2.8.6) (2.8.30).

8. Total line station labor-hours for deferrable unscheduled maintenance by LRU part number (2.8.8).

9. Accumulated delay time by airplane model (2.16).

10. Total number of cancelled nonrevenue flights of subject airplane at each line station (2.19).

11. Total number of cancelled revenue flights of subject airplane at each line station (2.19).

12. Number of departures by airplane model for each line station (2.20).

13. Total number of departures by line station (2.20).

14. Ending time of the simulation segment or run (SIMSCRIPT utilities).

15. Parts shortage tallies by LRU part number at each line station (3.5.4).

16. Total labor-hours expended for scheduled maintenance by LRU part number (4.1).

17. Total number of vendor repairs by component part number (5.1).

18. Number of failures by SRU part number (6.22)

19. Total repair time by SRU part number (6.22)

20. Total repair time for maintenance shop (6.22)

21. Total repair time by airplane model (6.22 and 6.31)

22. Total repair time for subject FCS (6.31)

23. Total quantity repaired by LRU number (6.35).

24. Total transit time through repair shop by LRU number (6.35).

5.9 PROCESSING AND STATISTICAL ANALYSIS

9.1 Starting the CEM

There are two general approaches to starting simulations such as the CEM:

- Start in a perfect state and run until the simulation stabilizes.

- Start with a best guess of steady state, having some failures present in airplanes, some stock levels depleted, and existing queues and operate the simulation until it stabilizes.

With either approach, the end state of the initializing run becomes the initial condition of the production runs. The first approach will use relatively more computer time while the second will require additional analyst time to synthesize the information carried in the processing data bases defined in Section 7.2. Development of an economical means of starting the simulation awaits the completion of its programming and the accrual of some operating experience.

A series of preliminary runs using different random number seeds are necessary to verify that the overall results are independent of starting conditions. This property is termed "stationarity." A lack of stationarity will necessitate adjustment of input data and reinitialization of the simulation since valid statistical inferences cannot be made for nonstationary stochastic processes. Stationarity is a necessary condition for testing whether or not a case is worthy of further consideration.

The analyst may need to extract a variety of information to determine both stationarity and whether the case under analysis is worthy of further consideration. This information may include items such as:

- Degree of agreement between spares outage probabilities and those used in the AOM

- Queue lengths at line stations and the airline repair shop

- Ratio of slack time to total labor-hours
- Ratio of overtime to total time
- Cancellation rate
- Delay time accumulation
- Quantity of AOGs
- Server activity level

9.2 Stopping the CEM

A simulation run, regardless of its length, will yield only one life-cycle cost estimate. Therefore, replication of simulation runs will be required if the statistical properties of life-cycle costs, returns on investment, and other economic variables are required.

An attractive artifice to increase the yield of cost data is segmentation, in which cost data are collected at evenly spaced, simulated time intervals during simulation runs and then scaled to yield annual costs. This process increases variance and may introduce bias but will appreciably reduce the running time that would otherwise be required.

Until CEM operating experience is accumulated, it would be premature to specify the most appropriate stopping rules; however, Fishman's Autoregressive Method (ref. 8), appears promising. Reference 8 also includes a SIMSCRIPT-coded subroutine for performing autoregressive analysis.

A standard problem is determining the sample size needed to estimate proportions such as cancellation rates to a specified accuracy and statistical confidence level. Employing the normal approximation to the binomial distribution, the general relationship for sample size is given by

N = P * (1-P) * (Z/E) ** 2

where

N = number of simulated flights required
P = a prior estimate of the proportion of interest
E = tolerance of the estimate or error (5-1)

Z = standard normal variate so that

$$\int_{-\infty}^{Z} N(0,1)\, dZ = 1 - ALPHA/2$$

where

ALPHA = 1 - (confidence level expressed as a decimal)

N(0,1) = standard normal probability density with mean = 0, standard deviation = 1

Values of Z versus confidence level are shown as:

CONFIDENCE LEVEL	Z
0.99	2.576
0.95	1.960
0.90	1.645
0.80	1.282
0.60	0.842

If E, in equation 1 is expressed in terms of percent error in P, then the final relationship becomes

$$N = 10\,000\,\frac{(1-P)}{P}\left(\frac{Z}{\text{PERCENT ERROR}}\right)^2 \qquad (5\text{-}2)$$

Table 3 provides N-values for various proportions, percent accuracies, and confidence levels.

ORIGINAL PAGE IS
OF POOR QUALITY

Table 3. Number of Simulated Flights Required to Estimate Proportions Versus Confidence and Accuracy

		ACCURACY, %				
PROPORTION	CONFIDENCE, %	2	5	10	20	50
0.0001	99	165 877 811	26 540 450	6 635 112	1 658 778	265 404
	95	96 030 396	15 364 863	3 841 216	960 304	153 649
	90	67 643 860	10 823 018	2 705 754	676 439	108 230
	80	41 083 991	6 573 439	1 643 359	410 840	65 734
	60	17 722 328	2 835 572	708 893	177 223	28 366
0.001	99	16 572 850	2 651 655	662 914	165 729	26 517
	95	9 594 396	1 535 103	383 776	95 944	15 351
	90	6 758 297	1 081 328	270 332	67 583	10 813
	80	4 104 701	656 752	164 188	41 047	6 568
	60	1 770 638	283 302	70 826	17 706	2 833
0.01	99	1 642 355	262 777	65 694	16 424	2 628
	95	950 796	152 127	38 032	9 508	1 521
	90	669 741	107 159	26 790	6 697	1 072
	80	406 772	65 084	16 271	4 068	651
	60	175 469	28 075	7 019	1 755	281
0.05	99	315 199	50 432	12 608	3 152	504
	95	182 476	29 196	7 299	1 825	292
	90	128 536	20 566	5 141	1 285	206
	80	78 067	12 491	3 123	781	125
	60	33 676	5 388	1 347	337	54

6.0 CASH FLOW ANALYSIS

This section contains all the algorithms required to produce, evaluate, or compare the cash flows for a given configuration of the fault-tolerant flight control system (FTFCS) based on statistics generated by the Analytical Optimization Method (AOM) and the Comprehensive Evaluation Method (CEM). In addition, a comparison of cash flows for different system configurations can be made that permits calculations of three figures of merit; namely, present equivalent cumulative cash flows, payback points, and return on investments. To arrive at these figures of merit, the total costs and benefits for each alternative is determined as the result of procuring, operating, obtaining benefits from, and disposing of a product. They are expressed as follows:

TCB = IC + OC + TA + RCC + OB

where

TCB = total costs and benefits
IC = investment costs
OC = operating costs
TA = tax adjustments
RCC = retirement costs and credits
OB = operating benefits

The convention is adopted that money received is positive (+), and money paid out is negative (-).

The steps in performing the economic analysis are:

Step 1. Using the details for given FTFCS configurations, calculate the cash paid out or received for each year the equipment is owned.

Step 2. Calculate the costs and revenues for each year, keeping each year separate.

Step 3. Calculate the present equivalent value for each year's payments and receipts from the formula:

PEVTCB (J) = TCB(J)/(1 + MARR(J)/100)**J

where

MARR = percent minimum acceptable rate of return (the rate that just meets the airline's threshold of acceptability)

PEVTCB(J) = present equivalent value of all payments, benefits, and receipts in the Jth year of operation

TCB(J) = total costs and benefits for year J

J = number of years from start of operation (J = 0 is the first year of investment and operation)

PRECEDING PAGE BLANK NOT FILMED

Step 4. Steps 1 through 3 are repeated for each design alternative. If subscript K is used to denote the design alternative (e.g., K = 1 is the first design scheme, K = 2 is the second), then

PEVTCB (J,K) = the present equivalent value of all payments, benefits, and receipts in the Jth year for the Kth design alternative

PEVIC(J,K) = the present equivalent value of investment costs made in the Jth year for the Kth design alternative

PEVIC(J,K) = IC(J,K)/((1 + MARR(J)/100)**J)

Step 5. As a basis for comparison, choose the design alternative with the lowest present equivalent value of investment cost. For this design alternative, let K = KMIN. For the case where cash benefits are not separately identified, the costs of each scheme can be compared. If the scheme with the lowest present equivalent value of total investment cost, PEVIC, is denoted by putting K = KMIN, then:

$$\mathrm{EROI(NY,K)} = 100\left\{\left[\frac{\sum\limits_{J=0}^{NY} \mathrm{PEVTCB(J,KMIN)} - \sum\limits_{J=0}^{NY} \mathrm{PEVTCB(J,K)}}{\sum\limits_{J=0}^{NY} \mathrm{PEVIC\ (J,K)} - \sum\limits_{J=0}^{NY} \mathrm{PEVIC\ (J,KMIN)}} + 1\right]^{\frac{1}{NY}} - 1\right\} \quad (1)$$

where

EROI (NY,K) = the extra return on investment (the amount by which return on investment [ROI] exceeds MARR) through a period of NY years (expressed as a percent) for scheme K compared with the scheme requiring the minimum investment (scheme KMIN)

Note that if the numerical value of the inner term of this equation is between 0 and 1.0, the equivalent return on investment (EROI) is negative. If the inner term is negative, the NYth root is imaginary. For both these cases, the EROI (NY,K) must be calculated from the expression below where NY is a positive integer. That is,

$$\mathrm{EROI(NY,K)} = -100\left\{\left|\frac{\sum\limits_{J=0}^{NY} \mathrm{PEVTCB(J,KMIN)} - \sum\limits_{J=0}^{NY} \mathrm{PEVTCB(J,K)}}{\sum\limits_{J=0}^{NY} \mathrm{PEVIC\ (J,K)} - \sum\limits_{J=0}^{NY} \mathrm{PEVIC\ (J,KMIN)}} - 1\right|^{\frac{1}{NY}} - 1\right\} \quad (2)$$

The primary reason for using present equivalent values in equations (1) and (2) is to account for differences in cash flow timing. For example, if two fault-tolerant

system (FTS) alternatives with identical total costs were compared, then the expression for EROI differentiates between one FTS with evenly spread operating costs as opposed to one with most of the operating costs late in its operating life. The latter case has the highest EROI.

EROI is based on the assumption of a symmetrical relationship for positive or negative ROI for the same absolute value of cash flow. Equations 1 and 2 also can be used to determine the internal rate of return (equal to the value of MARR for which the EROI = 0). In addition, the formula can be used to determine the net terminal return on investment by making MARR = 0. With a comparison of independent alternatives, those with the highest EROI are chosen until an investment of the desired size has been made. An example of independent alternatives might be the use of separate systems for wing load alleviation or flutter mode control or latteral augmented stability control. For mutually exclusive alternatives, for instance making a decision to use SIFT or FTMP in a given system, the preferred scheme is the one with an EROI greater than 0 and the highest value of net terminal cash. That is,

$$\sum_{J=0}^{NY} \mathrm{PEVTCB}\ (J,KMIN) \quad - \sum_{J=0}^{NY} \mathrm{PEVTCB}\ (J,K)$$

When cumulative benefits are separately identified for each case, then an EROI can be calculated using equations 1 and 2, with all terms involving Scheme K set to zero.

Step 6. The payback point (PP) is calculated by incrementing J from its starting value until the year JX is found in which the EROI changes sign from its last negative value.

$$PP = (JX - 1) + \frac{-\mathrm{EROI}\ (JX - 1)}{\mathrm{EROI}\ (JX) - \mathrm{EROI}\ (JX - 1)}$$

where

PP	=	payback point in years from the start of operation
JX	=	the year in which the extra return on investment changes sign from its last negative value
EROI (JX - 1)	=	the last negative value of EROI
EROI (JX)	=	the next positive value after the (JX -1)th value of EROI. (At the start of the first year of operation, the EROI = -100%.)

It is possible that, within the study period (NY), there are multiple payback points. If the maximum EROI does not occur after the last payback point, a study should be made to determine replacement costs and a replacement strategy.

ORIGINAL PAGE IS
OF POOR QUALITY

6.1 INVESTMENT COST

Investment cost (IC) is the cost of all properties and funds required for an airline to set up a business. ICs needed for FTFCS evaluation consist of:

$$IC = ICAP + ICRS + ICES + ICGS + ICST + ICTM + OTHER$$

where

ICAP =	airplane parts procurement and installation
ICRS =	rotable spares investment
ICES =	expendable spares initial stock
ICGS =	ground support equipment
ICST =	special tools and test equipment
ICTM =	training equipment

Investment cost data for rotable spares and expendable spares in terms of quantities required are output from the AOM and CEM respectively. The user will be required to provide the cost per rotable or expendable item. Other investment costs are user-supplied data.

All investment costs provided by the user are to be in first-year-of-operation dollars because the model provides only for a steady-state fleet size.

6.1.1 Airplane Parts Procurement and Installation (ICAP)

The cost of procuring and installing parts on an airplane consists of the price charged by the parts supplier and the installation costs, both multiplied by a profit markup factor for the airplane manufacturer.

In addition to the profit markup, new airplanes are subject to a 5% progress payment schedule for each of seven quarters before delivery. Thus, the prepayments have the effect of increasing the airplane price by a factor of 1.06 when progress payments are converted to a present value at 15% MARR.

The installation cost per part, in first year of operation dollars, is provided by the user after which it is multiplied by the quantity per airplane from the AOM or CEM, the profit markup, and prepayment factors to arrive at ICAP.

6.1.2 Rotable Spares Investment (ICRS)

The cost of rotable spares is the product of the cost per unit in first-year-of-operation dollars times the number of each unit obtained as output from the AOM or as determined by the model user.

6.1.3 Expendable Spares Initial Stock (ICES)

It is necessary to invest in a sufficient stock of expendable spares to take care of periods of heavy demand. An empirical method (ref. 9) used by several airlines has been used as follows:

Step 1. Using data collected from the CEM, calculate the number of spares of a given type expended in a year and determine the annual usage cost by multiplying the quantity expended by the cost per component (in 1977 dollars so that step 2 can be taken).

Step 2. Using the annual cost from step 1, enter Table 4 and determine the number of months supply to be stocked.

Table 4. Material Stock Levels

ANNUAL COST (1977 DOLLARS)	NUMBER OF MONTHS OF SUPPLY
0 to 199	12
200 to 499	6
500 to 999	4
1000 to 3000	2
Over 3000	1

Step 3. Arrive at the investment cost, ICES, by applying the following formula:

$$ICES = MM \times NMS/12$$

where

MM = annual cost per fleet

NMS = number of months supply

Step 4. Reinflate ICES from 1977 to base year dollars.

Table 4 is based on Reference 10 and considers the cost of replenishment and the cost of holding stock to determine an economic order quantity. If a significant number of FTFCS parts turns out to be expendable, a more appropriate algorithm would have to be developed.

6.1.4 Ground Support Equipment (ICGS)

The cost of ground support equipment includes the procurement cost of stands, slings, jigs, fixtures, tools, gages, jacks, servicing rigs, test equipment, vehicles, and anything used for maintaining, overhauling, repairing and testing airplanes, engines, and rigging flight controls. Such items are designed for use with any airplane type, or they become special items and should be included in ICST below. Thus, ICGS will include general-purpose automatic test equipment (ATE).

6.1.5 Special Tools and Test Equipment (ICST)

The cost of special tools and test equipment includes equipment that can be used only on one airplane or equipment type. General-purpose equipment is to be included in ICGS. For example, flight control rigging devices are included in ICST.

6.1.6 Training Equipment (ICTM)

This is the investment cost in training equipment for such items as students' notes, models, movies, and training aids. Flight simulators may require modification for different configurations of fault-tolerant systems, and in such a case modifications are included as a part of ICTM.

6.2 OPERATING COST

Input data obtained from the AOM or CEM for determining operating cost for a given system are noted in the subsections describing the various cost algorithms. Other input data are the user's responsibility.

Operating costs (OC) consist of all costs associated with airline operation. The definitions below identify cost categories that are provided for, and it will be noted that costs such as spares holding, delays, and cancellations do not correspond to the Civil Aeronautics Board (CAB) Form 41 cost breakdown. The OCs are defined as the sum of the cost entities below:

$$\begin{aligned} OC = \; & MLL + MSL + MM + SSC + MB + OS + MT + FCT + SH + FCC + FCP \\ & DC + CN + DT + CDS + CLP + TCE + TCP + OTHER \end{aligned}$$

where

- Labor-related operating costs

  MLL = maintenance line labor
  MSL = maintenance shop labor
  MM = maintenance materials
  SSC = shop and servicing supplies
  MB = maintenance burden

- Other operating costs

  OS = outside services
  MT = maintenance training
  FCT = flight crew training
  SH = spares holding cost
  FCC = fuel cost change
  FCP = fuel cost penalties
  DC = delay costs
  CN = cancellation costs
  DT = diversion and turnback costs
  CDS = debt servicing
  CLP = lease payments
  TCE = equipment transportation costs
  TCP = personnel transportation costs

Inflation of any operating costs not specified by means of a lookup table will be at a standard user-supplied inflation rate (default = 8%).

Labor-Related Operating Costs. MLL, MSL, MM, SSC, and MB are discussed below.

6.2.1 Maintenance Line Labor (MLL)

Maintenance line labor cost consists of the compensation paid to all personnel engaged in line maintenance plus the employee insurance, fringe benefits, and pensions that are not directly included in compensation. Maintenance line labor is to be calculated as:

$$MLL = \sum_{J=0}^{RAGE} ML\ (J) + (MLOT(J) \times MLOR) \times BPMH\ (J)$$

$$ML\ (J) = \sum_{S=1}^{NS} (MHL\ (J,S) \times SLF\ (S))$$

$$MLOT\ (J) = \sum_{S=1}^{NS} (MHLOT\ (J,S) \times SLF\ (S))$$

where:

MLL	=	maintenance line labor cost in dollars for all J years summed from J = O to the retirement age (RAGE) in years
ML(J)	=	maintenance line labor for the Jth year for regular time in hours (CEM output)
NS	=	the number of skill levels (usually one)
MLOT(J)	=	maintenance line labor overtime for the Jth year (CEM output)
BPMH(J)	=	base pay/labor hour for year J (table 5), dollars
MLOR	=	ratio of overtime to base pay BPMH(J)
MHL(J,S)	=	labor hours line in the Jth year for the Sth skill level obtained as output from the line maintenance simulation. MHL(J,S) for fractions of a year must be multiplied by 365/days simulated.

SLF(S) = skill level compensation ratio for skill level S of NS

$$= \frac{\text{compensation for skill level, S(J=0) (supplied by the user)}}{\text{compensation base rate, CPMH(J=0) (from table 5)}}$$

MHLOT(J,S) = overtime labor hours line in the Jth year for the Sth skill level obtained as output from the line maintenance simulation. MHLOT(J,S) for fractions of a year must be multiplied by 365/days simulated.

Table 5. Base Pay per Labor Hour

YEAR	BASE $/LABOR HOUR (BPMH)
1971	5.97
1972	6.54
1973	7.10
1974	7.62
1975	8.46
1976	9.10
(1977)	9.99
(1978)	10.87
(1979)	11.64
(1980)	12.80
(1981)	13.75
(1982)	14.90

NOTE: Reference CAB Schedule P10 Form 41. Years in parentheses are estimates, since reporting stopped in the third quarter of 1977.

6.2.2 Maintenance Shop Labor (MSL)

Maintenance shop labor is calculated in the same way as MLL, except that MHL(J,S) and MHLOT(J,S) are changed to MHS(J,S) and MHSOT(J,S) wherever they appear and are obtained from the repair shop simulation.

6.2.3 Maintenance Materials (MM)

Maintenance material is the total cost of maintenance materials plus expendable parts purchased in a given year to replace those consumed. Maintenance material usage is generated only in the simulated repair shops. Input to the CFA from the CEM will be directly in units of dollars material cost per fleet per year in a specified year's dollars. Inflation or deflation of specified year's dollar input will

be in accordance with Table 6, which can be extended using Chase Econometric Service or other econometric surveys:

Table 6. Material Inflation Factors

YEAR	INFLATION FACTOR	MATERIAL INCREASE, %
1969	0.6683	4.4
1970	0.7090	6.1
1971	0.7289	2.8
1972	0.7500	2.9
1973	0.7905	5.4
1974	0.8696	10.0
1975	1.0000	15.0
1976	1.0750	7.5
1977	1.1309	5.2
1978	1.2282	8.6
1979	1.4026	14.2
1980	1.5485	10.4
1981	1.6243	4.9

6.2.4 Shop and Servicing Supplies (SSC)

Shop and servicing supplies cover the cost of supplies and expendable small tools and equipment used in maintaining, servicing, and cleaning property and equipment that cannot be directly assigned to a specific job or type of work. Because a cost-estimating relationship is not available for SSC, the analyst must estimate it from his knowledge of the equipment to be serviced.

6.2.5 Maintenance Burden (MB)

Maintenance burden (or overhead) is a total airline system-related cost that has been allocated back to airplane types. It is not an airplane- and engine-originating cost like fuel consumption or direct maintenance material consumption, and it is not a property of an airplane type as reported to the CAB.

Total maintenance costs are divided according to CAB Form 41 into three direct charge accounts for airframe, engines, and other. An indirect account, burden, is further subdivided into a number of accounts, consisting of:

- Labor for ground property and equipment
- Material for ground property and equipment
- Maintenance trainees and instructors
- Unallocated labor
- Communications personnel
- Recordkeeping and statistical personnel
- Purchasing personnel
- Other personnel
- Utilities (heat, light, power, water)
- Outside services
- Rentals
- Shop, servicing supplies
- Employee benefits
- Payroll taxes

CAB-reported burden is, in fact, an inseparable mixture of airline-sensitive and airplane-sensitive elements. Airline-sensitive elements include a very large number of independent and interdependent elements, among them:

- The mix of airplane types
- Route structure
- Geography and climate
- Maintenance philosophy
- Labor union contractual provisions
- Efficiency
- Management infrastructure

To compound the analytical problem, a great deal of latitude is inherent in CAB reporting requirements and as a result, tremendous differences exist among various airlines flying identical equipment. As deregulation continues, even this flawed (from the standpoint of making airplane comparisons) data base is likely to disappear. Given this environment, it would be tempting to ignore burden altogether. Yet, to do so would bias comparisons. For example, a maintenance scheme that relies on rotable spares and is, therefore, labor-intensive, will not be correctly compared with one that relies upon replaceable spares and is material intensive. In other words, one recognizes that there is a design-sensitive burden to be compared among designs, and this entity is what Cost and Benefit Optimization Model (CBOM) attempts to handle. Design-sensitive burden, then, and CAB Form 41 burden are distinct entities. The former is appropriate to design comparisons; the latter is useful for assessing financial aspects of particular airline operations. Users of CBOM are cautioned that these two types of burden are related only because they share some common terminology. This does not preclude the use of CAB data inferences where appropriate, such as labor fringe benefit factors and the ratio of support personnel to direct labor.

Design-sensitive burden includes two major elements:

- Labor burden
- Labor and material for maintenance of ground property and equipment

Spares holding cost, outside services, and shop and servicing supplies, which also are design-sensitive and normally included as burden in CAB Form 41, are separated out in the CBOM under SH, OS, and SSC.

Labor burden elements consist of:

- Payroll Taxes. Federal payroll taxes (FICA) will be applied to direct wages at the 1979 rate of 6.13% escalating at an additive rate of 0.1%/year after 1979. The state rate will be computed at 50% of the federal rate. For a composite rate of 9.2%, escalating at 0.15%/year, the multiplicative factor is 1.092 + 0.0015/year after 1979.

- Fringe Benefits According to CAB Statistics. The 1979 fringe benefit factor for insurance and pensions is 1.23 x direct wages, escalating at 1% per annum, additive.

- Nonproductive Time. The ratio of total paid time per year to productive hours worked is 2080/1870 = 1.113 and accounts for vacations, sick leave, and holidays.

- Support Personnel (Guards, Custodians, Tool Crib Attendants). The best estimate of this is obtained from CAB Form 41 data that show unallocated shop labor is equal to 20% of total burden. Because the average ratio of total burden to direct labor is 2.7:1, this category is equal to 2.7% x 20% = 54% of direct labor. Therefore, multiplicative factor is 1.54.

- Administration (Timekeeping, Payroll). Administrative costs are estimated as 0.5% of payroll.

For 1979, the overall multiplicative factor applied to direct labor is

$$1.092 \times 1.23 \times 1.113 \times 1.54 \times 1.005 = 2.314$$

For 1980, the overall factor is

$$(1.092 + 0.0015) \times (1.23 + 0.01) \times 1.113 \times 1.54 \times 1.005 = 2.335$$

Labor and Material for Maintenance of Ground Property and Equipment. The labor component comes to 3.75% of total burden, which averages 2.7 times direct labor; 2.7 x 3.75% = 10.1% of direct labor. This figure is supplied to guide the analyst who must input this cash flow for FTFCS into the CBOM.

6.2.6 Outside Services (OS)

Outside services will be used as a separate cost category for FTFCS equipment repaired by an associated or nonassociated company. Input to the CFA will be from the CEM in terms of ORPY, the number of total outside repairs by year; ORMH, the average outside repair man (labor) hours per repair; ORMM, the average outside material cost per repair; and the year dollars associated with the material cost. OS is computed as follows:

OS(J)	=	((ORMH x BPMH(J) x OSB) + (ORMM(B) x MMF(J) x ORPY(J)) x OSPM)

where

OS(J)	=	outside services cost in dollars for year J for the fleet
ORMH	=	average outside repair man (labor) hours per repair
BPMH(J)	=	base pay per man (labor) hour (table 5) for year J
ORMM(B)	=	average outside repair material cost per repair for the user-specified base year B
MMF(J)	=	inflation/deflation factor to convert maintenance material costs from year B to year J. Factors are provided in Table 5.
ORPY(J)	=	the number of outside repairs for the fleet in year J from the CEM
OSB	=	outside services burden factor (default = 2.335 for 1980) (sec. 6.2.5)
OSPM	=	outside services profit markup factor (assumed default = 1.15. Further work is required to validate this value.)

6.2.7 Maintenance Training (MT)

Maintenance training consists of nonrecurrent and recurrent training associated with the introduction of new equipment. An approximate method of estimating maintenance training cost is to use $10 per student hour (1982 dollars).

6.2.8 Flight Crew Training (FCT)

Flight crew training consists of nonrecurrent and recurrent training associated with the introduction of new or modified airplanes. A method of estimating flight crew training cost has not been developed and will be determined by the user if needed.

6.2.9 Spares Holding Cost (SH)

Spares holding cost is the annual cost of holding rotable and expendable spares and materials in stock, consisting of:

- Warehousing
- Recordkeeping
- Administration of stocks and stores
- Inventory taxes
- Insurance

Spares holding cost can be estimated from the formula:

$$SH = SHP \times (ICRS + ICES) \times MMF/100$$

where

SH	=	spares holding dollars per year per fleet
SHP	=	spares holding cost percentage of inventory
	=	10% (a user override for SHP will be provided for the CBOM)
ICRS	=	rotable spares investment
ICES	=	expendable spares and material investment
MMF	=	maintenance material inflation factor

In the above expression, SHP (the holding cost as a percentage of inventory) is based on an industry-accepted figure of 25%, which includes cost of capital. Since cost of capital is accounted for in the CBOM by using present equivalent value accounting with an MARR of 15%, the residual holding cost is 10%. Of this 10%, approximately 25% is recordkeeping and administration. Recordkeeping and administration are included in maintenance burden in CAB Form 41 reports, but for design analysis they have been separated out as a function of spares inventory value. Further work is required to verify the industry-accepted spares holding costs.

6.2.10 Fuel Cost Change (FCC)

Fuel cost change due to eliminated weight or drag is to be separated from fuel cost penalties (FCP) resulting from component failures. Incremental fuel saved or burned is determined by the CBOM user (or his aerodynamicist) using standard performance calculations based on airplane aerodynamic and engine performance data in units of weight of fuel burned per unit of incremental weight change per flight hour. Typical mission summaries are shown in Table 7, and the resultant incremental reduction or cost of weight are provided in Table 8.

Table 7. Mission Summary

Model	707-32-0B	727-100	727-200	737-200	747-200	747SP
Engines	JT3D	JT8D-7	JT8D-9	JT8D-9	JT9D-7A	JT9D-7A
Average flight (hr)	3.23	1.16	1.11	0.80	5.38	7.55
Average flight (km)[a]	2 576	793	734	526	4 637	6 618
Payload (kg)	8 437[b]	5 307[b]	7 575[b]	6 214[b]	44 920[c]	30 402[c]
Reserves (kg)	6 468	4 536	4 536	3 175	16 284	14 515
OEW (kg)	64 864	40 370	45 359	28 576	170 006	144 923
Fuel consumed per flight (kg)	13 376	3 931	4 073	1 960	57 565	65 771
Brake release gross weight (kg)	93 145	54 143	61 543	39 925	288 485	255 372
Climb speed schedule						
U.S. rules KEAS/Mach	No 300/0.78	Yes 280/0.75	Yes 280/0.75	Yes 280/0.65	Yes 320/0.81	Yes 320/0.81
Cruise Mach	0.78	0.80	0.80	0.72	0.84	0.85
Cruise altitude (m)	11 887	10 668	10 668	9 144	10 668	12 497
Descent speed schedule						
U.S. rules	No	Yes	Yes	Yes	Yes	Yes
KEAS/Mach	260/0.78	280/0.80	280/0.80	280/0.75	320/0.81	320/0.81
Temperature	Standard day	Standard day	Standard day	Standard day	Standard day	Standard day
Winds	0	0	0	0	0	0

[a]Based on scheduled carrier data, cumulative through July 1974 for 707, 727, 737 and based on September 1974, September 1975, and September 1976 for the 747

[b]Nominal 55% passenger load factor + cargo

[c]55% load factor with volume limit of cargo

Table 8. Cost of Additional Airplane Weight, Based on 3000 Flight Hours per Year

Airplane model	707-320B	727-100	727-200	737-200	747-200	747SP
Kilograms of fuel/flight hr/ kg of weight	0.05636	0.03707	0.03964	0.04688	0.04978	0.04969
Weight of fuel/ additional weight/yr(kg)	169.1	111.2	118.9	140.6	149.4	149.1
Cost of 1.0 kg weight/year						
30¢/gal	16.69	10.98	11.75	13.89	14.75	14.75
40¢/gal	22.26	14.64	15.65	18.52	19.66	19.66
50¢/gal	27.82	18.29	19.58	23.13	24.58	24.58
\$1/gal	55.64	36.58	39.16	55.52	58.98	58.98

The fuel used/flight hour in Table 8 is accurate only for the average flight of Table 7. The exact determination of error in applying fuel used/flight hour based upon average flights to shorter or longer flights has not been established.

FCC is calculated as shown below.

$$FCC = FCW + FCD$$
$$FCW = FCPA \times WIC \times UTIL \times NA \times DG/PG$$

where

FCC = fuel dollars change/fleet/year
FCW = fuel dollars change/fleet/year due to weight
WIC = weight change, kilograms
UTIL = utilization in flight hours/airplane/year
NA = number of airplanes in the fleet
DG = fuel price, dollars/liter
PG = kilograms/liter of fuel (equals 0.8029)
FCPA = pounds of fuel consumed/pound of added weight (or saved/pound of reduced weight)/airplane/flight hour

Fuel consumed because of drag (FCD) can be derived from the expression below.

$$FCD = FCPD \times DIC \times UTIL \times NA \times DG/PG$$

where

FCD = fuel dollars change/fleet/year due to drag
DIC = drag percent change
UTIL = utilization in flight hours/airplane/year

NA = number of airplanes in the fleet
DG = fuel price, dollars/liter
PG = kilograms/liter of fuel (equals 0.8029)

FCPD, the fuel consumed/1% increase in drag in kilograms of fuel/flight hour, is as follows:

- 36.28 kg of fuel/flight hour for a 707-320B with 3.23 hours average flight length
- 24.04 kg of fuel/flight hour for a 727-100 with 1.16 hours average flight length
- 27.22 kg of fuel/flight hour for a 727-200 with 1.11 hours average flight length
- 20.41 kg of fuel/flight hour for a 737-200 with 0.8 hour average flight length
- 107.96 kg of fuel/flight hour for a 747-200 with 5.38 hours average flight length
- 88.00 kg of fuel/flight hour for a 747 SP with 7.55 hours average flight length

As with the formula for fuel burned due to added weight, the drag cost-estimating relationship is provided as an approximate guide. In studies where a more accurate answer is required or where drag represents a significant portion of total cost, a detailed performance analysis is required.

6.2.11 Fuel Cost Penalties (FCP)

Fuel cost penalties result from dispatch with components of an FTFCS stage inoperative or component failures in flight and,

$$FCP = \frac{AFBP}{100} \times FCPF \times \frac{UTIL}{AFLH} \times NA \times \frac{DG}{PG}$$

where

FCP = fuel penalty, dollars/fleet/year
FCPF = fuel consumed per flight from Table 6-4 (kg)
UTIL = utilization in flight hours/airplane/year
AFLH = average flight length (hours)
NA = number of airplanes in the fleet
DG = fuel price (dollars/liter)
PG = kilograms/liter of fuel (equals 0.8029)
AFBP = average fuel burn penalty caused by failures (percent per flight)

An example of the way in which AFBP is supplied by the model user is shown in Table 9 where the notation A = 2 means that stage A has two units in unfailed condition.

C-3

ORIGINAL PAGE IS
OF POOR QUALITY

Table 9. Penalty Relationships

CONDITION	BOOLEAN RELATIONSHIPS	PERCENTAGE OF INCREASE IN FUEL BURN
I	A = 5 and B = 3 or C = 4 and D = 5	+1
II	A = 4 and B = 2 and C = 3 and D = 4	+2
III	A = 3 and B = 1 and C = 2 and D = 4	+3
IV	A L.T.3 or B = 0 or C L.E 2 or D L.T 4	+10

Imagine that the CEM yielded the following preflight relative frequencies for these stages (table 10).

Table 10. Preflight Relative Frequencies

	NUMBER OF UNFAILED UNITS						
STAGE	6	5	4	3	2	1	0
A	0.10	0.50	0.20	0.20	0	0	0
B	X	X	X	0.70	0.20	0.10	0
C	X	X	0.15	0.65	0.20	0	0
D	X	0.20	0.80	0	0	0	0

Next, the probabilities of achieving these various conditions will be calculated:

Probability I = (0.50)(0.7) + (0.15)(0.20) - (0.35)(0.03) = 0.3695
Probability II = (0.20)(0.20)(0.65)(0.80) = 0.0208
Probability III = (0.20)(0.10)(0.20)(0.80) = 0.0032
Probability IV = 0 + 0 + 0 + 0 = 0.0000

Other Boolean combinations do not affect fuel burn in this example. The expected fuel burn increase is

0.3695(1) + 0.0208(2) + 0.0032(3) + 0(10) = 0.4207%

The postflight data are shown in Table 11.

ORIGINAL PAGE IS
OF POOR QUALITY

Table 11. Postflight Relative Frequencies

NUMBER OF UNFAILED UNITS

STAGE	6	5	4	3	2	1	0
A	0.05	0.40	0.30	0.15	0.08	0.02	0
B	X	X	X	0.50	0.30	0.19	0.01
C	X	X	0.05	0.70	0.20	0.05	0
D	X	0.10	0.85	0.03	0.02	0	0

From these postflight statistics, the following is obtained:

Probability I = 0.2040
Probability II = 0.0536
Probability III = 0.0048
Probability IV = 0.3639

where

expected fuel burn increase = 3.9646%
average fuel burn penalty = (3.9646 + 0.4207)/2
= 2.1926%

In the above example a low reliability system has been used to illustrate the process of preparation of input for economic analysis. Actual fuel penalties for FTFCS degradation are expected to be much less.

6.2.12 Delay Costs (DC)

Delay costs for the airplane are calculated in the CFA by evaluating three tangible costs consisting of:

- Passenger handling costs
- Extra crew costs
- Lost passenger revenue

The following method is proposed:

DC = (PHC + ECC + LPR) x SQA x DPC x ADM x UTIL x NA/(AFLH x 6000)

where

DC = delay cost dollars/year/fleet

PHC = passenger handling cost, dollars/seat delay hour

PHC(76) = 0.2171

ORIGINAL PAGE IS
OF POOR QUALITY

ECC	=	extra crew cost, dollars/seat delay hour
ECC(76)	=	2.442 - 0.0038 SQS
LPR	=	lost passenger revenue, dollars/seat delay hour
LPR(76)	=	(LF x (27.5689 AFLH - 1.373) x 0.8712 EXP(0.0454 - 0.2271 AFLH))/(1 + 1.3877 AFLH)
SQS	=	seat quantity, standard for airplane type (table 12)
LFR	=	load factor (decimal, not percent)
SQA	=	seat quantity, actual
DPC	=	delays/100 flights, from the CEM
ADM	=	average delay time/delay (minutes), from the CEM
UTIL	=	utilization in hours/year/airplane, from the CEM
NA	=	number of airplanes in the fleet
AFLH	=	average flight length (hours), from the CEM

Table 12. Standard Airplane Seating

AIRPLANE	STANDARD SEATING (SQS)
737-200/DC9-40	115
727-200	131
DC10-10	270
L-1011	268
707/DC8	143
747	385

Implicit in the formula for DC are the relationships:

S = (AFLH - 0.2)/1.93
d = 0.4277 + 0.5867 x AFHL

where

S = flight length in thousands of statute miles
d = hours after which a delay becomes a cancellation

Inflation and deflation factors necessary to convert delay costs to other years are provided in Table 13 and are derived from CAB Form 41 reported pilots and copilots pay (Account 23) for major domestic carriers.

Table 13. Pilot and Copilot Pay Inflation Factors

YEAR	FLIGHT CREW FACTOR
1967	0.4847
1968	0.5485
1969	0.5960
1970	0.6570
1971	0.7040
1972	0.7370
1973	0.7737
1974	0.8436
1975	0.9439
1976	1.0000

Inflation factors for 1977 and on are calculated as follows:

FCF(J) = flight crew inflation factor for the Jth year of operation
FCF(J) = (1 + (FCINF/1000))**(YEAR + J - 1976)

where

FCINF = flight crew annual inflation rate as a percentage (8% default)
YEAR = calendar year for the start of operation

6.2.13 Cancellation Costs (CN)

Cancellation costs consist of all the costs of a delay up to the time the flight is cancelled plus costs associated with loss of airplane use for the flight hours it is out of service. It has not been possible to develop a method of allowing for airplane repositioning costs following a cancellation.

Calculation of the delay cost portion of cancellations is based on the average delay time preceding a cancellation, ADMC.

CN = (CNDC + CNDL) x CNPM x UTIL x NA/(1000 x AFLH)

where

CN = cancellation dollars/year/fleet
CNDC = cancellation delay, dollars/cancellation (see below)
CNDL = cancellation downtime, dollars/cancellation (see below)
CNPM = cancellations/1000 departures/airplane
UTIL = utilization, flight hours/year/airplane
NA = number of airplanes in the fleet
AFLH = average flight length in hours

- Cancellation, Delay Cost Contribution. For the above expression for CN:

  $$CNDC = (PHC + ECC + LPR) \times SQA \times ADMC/60$$

  where

  ADMC = average delay minutes preceding cancellation obtained by CEM simulation

  See DC (delay cost) for all other quantities.

- Cancellation Downtime Loss Contribution. For the above expression for CN:

  $$CNDL(1972) = 0.003 \times OEW \times FHL$$

  FHL = flight hours lost, obtained by simulation

  It should be noted that the above does not include the costs of eliminating problems that cause the cancellation. Such problems will be included in maintenance labor and material, ML and MM.

6.2.14 Diversion and Turnback Costs (DT)

Diversions and turnbacks will be calculated as follows:

$$DT(1977) = 0.0067 \times (517\ AFLH - 103) \times SQA \times DTY \times NA$$

where

DT = diversions and turnback dollars/year/fleet (1977 dollars)

AFLH = average flight length, hours

SQA = seat quantity/airplane, actual

DTY = number of diversions and turnbacks/airplane/year obtained from postflight system states produced by the CEM

NA = number of airplanes in the fleet

6.2.15 Debt Servicing (CDS)

The cash cost of servicing, obtaining, and repaying debt is to be included in the CFA. CDS includes all cash flows associated with debt; namely, receipt of a sum at time J = 0 equivalent to all investments made, interest payments (CIP) on the debt, and repayment of debt in the final year J = DEBTL. CIP is deducted from income before taxes, and its present equivalent value calculated. The income from unallocated debt funding will be neglected since it is nonexistent with a mature fleet size at the J = 0 year and small for the more realistic fleet build-up case. For simplicity, a single debt repayment is assumed to be effected in the DEBTLth year.

The following inputs to the CFA are required:

DEBTL = the term of the debt in years. The retirement age of equipment (RAGE) is to be used as a default of DEBTL.

DEBTI = the percentage annual interest paid on the debt (default = 12)

Interest payments in the Jth year are given by:

CIP = ICSUM x DEBTI/100

where

$$ICSUM = \sum_{J=0}^{RAGE} (ICAP(J) + ICRS(J) \ldots\ldots\ldots)$$

$$CDS = \sum_{J=0}^{DEBTL} CDS(J)$$

where

ICSUM = sum of all investment costs from Section 6.1
CIP = interest payments (of equal size) for each year
CDS(J) = debt cash flow in year J
CDS(J) = -ICSUM + CIP; for J = 0
= CIP; for J = 1 to (DEBTL - 1)
= ICSUM + CIP; for J = DEBTL

Present equivalent value of debt servicing cost PEVCDS(J) is calculated as follows:

$$PEVCDS(J) = \frac{CDS(J)}{(1 + MARR(J) / 100) ** J}$$

$$PEVCDS = \sum_{J=0}^{DEBTL} PEVCDS(J)$$

CDS = cumulative debt servicing cost

6.2.16 Lease Payments (CLP)

This is the negative cash flow of lease payment, made for a defined period of time by a lessee airline operator to the lessor who is the actual owner of the equipment. All investment tax credits and depreciation are to the benefit of the lessor;

therefore the lessee's payments are treated as a pure expense item that is deducted from the gross income (or savings) generated by the leased equipment. Because leases are not investments, competing lease schemes cannot be ranked using the investment criterion of EROI; instead the present values of the various alternatives should be used for ranking.

CLP is deducted from income before taxes, and its present equivalent value must be calculated.

For simplicity, only equal lease payments will be treated, and the value of purchase options will not be included. This simplification, however, corresponds to contemporary reality, in which variations on equal payments are seldom encountered.

The following additional inputs are required: annual lease percentage (ALP) (default 12) and the final year of the lease (FINL). While debt payments are made in arrears (i.e., after use of the money), lease payments are customarily made in advance.

The annual lease payment is a complex function of the lessor's cost of capital, the lessor's tax situation, the duration of the lease, the residual value of the equipment at lease end, and the requirements of lessor, lessee, and (frequently) the lender. In the 1982 business environment, a reasonable default value for a long-term lease will be ALP = 12% of the value of the leased item's ICSUM. Other percentage values may be input at the option of the analyst.

Lease payments in the Jth year are given by:

$$CLP(J) = APL/100 \sum_{J=O}^{RAGE} (ICAP\ (J) + ICRS\ (J)......)$$

$$CLP = \sum_{J=0}^{FINL} CLP(J)$$

where

CLP = cumulative lease payments
RAGE = retirement year of the FTFCS
FINL = final year of the lease

Present equivalent values of lease payments are calculated as follows:

PEVCLP(J) = CLP(J)/((1 + MARR(J)/100)**J)

ORIGINAL PAGE IS
OF POOR QUALITY

$$PEVCLP = \sum_{J=0}^{FINL} PEVCLP(J)$$

When lease is used in the CFA, the input will be made in the same manner as for investments. After ICSUM has been used to calculate CLP, the investment costs and associated investment tax and depreciation allowance will be zeroed out.

6.2.17 Equipment Transportation Costs (TCE)

Transportation costs for equipment are the costs for packing and shipping rotatables and components between stations and vendors.

TCE = SC + PC

where

TCE	=	transportation cost in 1979 dollars/shipment in the continental U.S.
SC	=	shipping cost (air freight)/one-way shipment, $35 minimum plus $0.4536/kg ($1/lb) for excess weight over 15.88 kg (35 lb), in 1979 dollars
PC	=	packing and unpacking cost at 30 minutes for each operation ($30, burdened 1979 dollars)

Packaged weight should be taken as component weight times 1.25. Component weight is a user input.

6.2.18 Personnel Transportation Costs (TCP)

For airlines that operate into stations with no maintenance resources, the cost of transporting mechanics when an airplane is grounded can be significant. Most frequently a charter flight is used, and the algorithms that follow are based on this assumption.

Transportation costs for personnel are the costs of flying mechanics to and from stations requiring support and are given by the expression:

TCP = (TFHC x RSFT) + (TSHC x TST)

where

TCP	=	cost per round trip, 1980 dollars
TFHC	=	charter flight cost per hour multiplied by jet-to-charter flight speed ratio of 4
TFHC	=	$400 (in 1980 dollars)
RSFT	=	round trip jet scheduled block time in hours (from the CEM)
TSHC	=	transportation standby cost in dollars/hour
TSHC	=	$20 (in 1980 dollars)
TST	=	transportation standby time in hours (from the CEM)

6.3 TAX ADJUSTMENTS

Tax adjustments (TA) that apply to FTFCSs may consist of three tax entities as shown below. The following paragraphs describe ITC, TDA, and INC.

TA = ITC + TDA + INC

For airlines that are not in a position to take advantage of tax credits because of inadequate income, a provision is required to eliminate all tax adjustments. The use of leasing eliminates ITC and TDA.

ITC. Investment tax credit for airplanes and capitalized equipment (except buildings) of 10% of the basis value may be deducted from tax that would otherwise be paid. The Economic Recovery Tax Act of 1981 provides details of tax incentives that have been incorporated in the following algorithms:

ITC = ITF x (ICAP + ICRS + ICGS + ICST + ICTM + OTHER)

where

ITF = investment tax credit factor (0.1 as of January 25, 1975)

ICAP, ICRS, ICGS, ICST, ICRE, and ICTM are defined investment costs. The amount of investment tax credit that can be claimed is limited to $25 000 + 90% of tax in excess of $25 000 during 1979, and the percentage increases each year. The assumption is made that sufficient tax is paid to take advantage of all credits as they occur except for the option detailed in TA above.

TDA. Tax depreciation allowance is provided under the Economic Recovery Act of 1981. Three depreciation schedules are furnished, and their selection depends on the year the asset is placed in service.

Table 14 shows the depreciation factors allowed by the Internal Revenue Service.

Table 14. Tax Depreciation Factors

YEAR OF RECOVERY	DATE PLACED IN SERVICE: 1981 TO 1984	1985	1986 AND ON
1	0.15	0.18	0.20
2	0.22	0.33	0.32
3	0.21	0.25	0.24
4	0.21	0.16	0.16
5	0.21	0.08	0.08

Property is depreciated to zero residual value and gains made on its disposal are treated as ordinary income. Each year's depreciation may be treated as an expense that is deductible from pretax income. Since corporate tax on U.S. income consists of 46% Federal taxes plus approximately 2% state taxes, the tax depreciation allowance is equivalent to a 48% credit of each year's depreciation. For design studies, tax depreciation allowance is calculated from the formula:

$$TDA = TDF \times 0.48 \times (ICAP + ICRS + ICGS + ICST + ICTM + OTHER)$$

where the tax depreciation factor, TDF, is obtained from Table 14 and ICAP can be obtained using the definitions in Section 6-1.

For those airlines unable to take advantage of the fastest allowable depreciation because of tax carryovers or anticipated losses, provision will be made for the user to provide his own values of TDF for up to 15 years.

Note also that ordinary and necessary expenditures paid or incurred during the year for repairs to depreciable property are allowable expenses and are deductible for the current year. Expenditures during the year that substantially prolong the life of the property, or that increase its value or adapt it to a different use, are ordinarily classified as capital expenditures and are recovered through annual depreciation deductions over the useful life of the property. For example, after 50 000 flight hours an airplane undergoes a major structural overhaul that extends its life for another 30 000 hours. The depreciated value of the airplane then would be increased by the cost of the work and treated as an investment under ICAP.

INC. Federal and state income taxes for design studies are at 48% of gross income less allowable expenses. It may be assumed that all costs included in OC (sec. 6.2) are allowable. Therefore, by subtracting allowable expenses before calculating income tax, the impact of operating costs is reduced and can be considered as equivalent to a decrease in costs and an increase of benefits.

ORIGINAL PAGE IS
OF POOR QUALITY

6.4 RETIREMENT COSTS AND CREDITS

Equipment retirement may be planned for the end of its useful life, or it may be premature as a result of obsolescence or failure. Both income and expenses may result from salvage and retirement. In the case where net retirement income exceeds the depreciated value of the asset, the excess is taxed as regular income.

$$NRC = RS + RP \text{ if } NRC \leq \sum_{i=1}^{J-1} TDA\,(i)$$

or

$$NRC = (RS + RP) \times 0.52 + 0.48 \sum_{i=1}^{J-1} TDA\,(i)$$

$$\text{if } NRC > \sum_{i=1}^{J-1} TDA\,(i)$$

where

NRC = retirement net dollars/fleet in the year of retirement

RS = retirement sales income, dollars/fleet for the year of retirement

RP = retirement preparation cost for overhaul, refurbishing, and inspection, dollars/fleet in the year of occurrence

$\sum_{i=1}^{J-1}$ TDA (i) = cumulative depreciation over J years to retirement

The algorithms for retirement credit are included for completeness and will not be incorporated in the CFA because their use for FTFCS is unlikely.

6.5 OPERATING BENEFITS

Positive cash flows produced by a given FTFCS are defined as operating benefits (OB). Positive cash flows might be generated as a result of:

- Increased range for a specified payload and vice versa
- Increased passenger appeal (and demand) as a result of improved ride quality and dispatch reliability
- Reduced fuel burn

Provision will be made for the analyst to input benefits of a specific fault-tolerant system configuration into the CBOM in units of dollars (of a specified year) per flight hour for flights of a given length. The extra return on investment, present equivalent value, and payback point for the benefits can then be calculated by treating benefits as positive costs and eliminating the set of terms involving K in formulas (1) and (2) of Section 6.0.

6.6 RISK

For the CBOM, risk is defined as the probability that the ROI is less than the MARR.

The ability to perform risk analyses will be provided for the combination of the Comprehensive Evaluation Model (CEM) and the CFA economic analysis. The method entails randomly varying component reliability, repair time, and purchase price about their average value to produce probability distributions for return on investment (ROI) and after-tax disposable income.

The above procedure might well produce the situation illustrated in Figure 35, where the configuration with the greatest ROI also has the greatest probability of being less than the MARR. The CBOM user must make the final selection.

The procedure of using the CEM and CFA to generate probability distributions such as those of Figure 35, is one of brute force. It remains to be seen, when the CEM is coded and run, if the computer time required makes risk analysis infeasible with the CEM in its present form. The use of one version of CEM for evaluation and a simplified version for risk analysis is a question to be answered by the next phase.

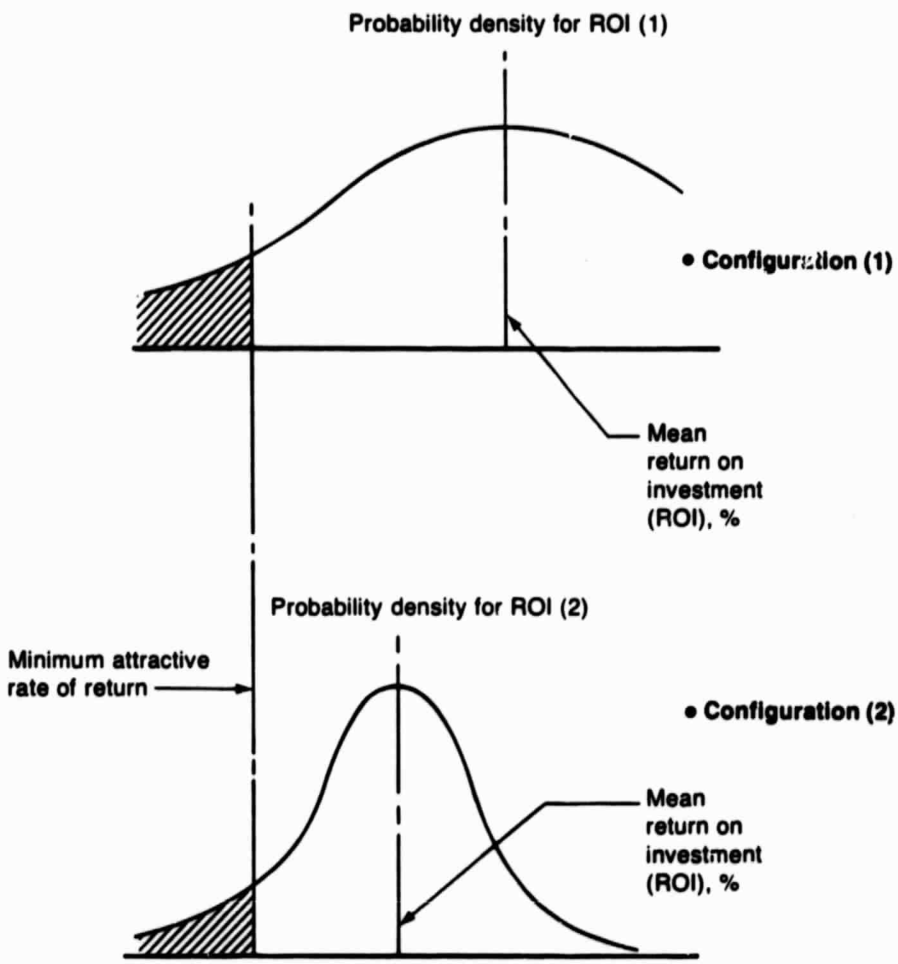


Figure 35. Risk Analysis

7.0 REFERENCES

1. Siebert, G. A., "Methodology for the Quantitative Optimization of Availability for Fault Tolerant Flight Control Systems Design: A Systems Approach," Boeing Computer Services Company Final Report D6-51795, December 1982.

2. Avizienis, A., "Fault Tolerance: The Survival Attribute of Digital Systems," IEEE, Vol. 66, No. 10, October 1978.

3. Beard, M. C., Director of Airworthiness, FAA, "Advisory Circular Draft," Initiated by AW3-130 and FAA "Advisory Circular 25.1309-1," 9/7/82, Initiated by ANM-110.

4. Noyce, Robert N., "Microelectronics," *Scientific American Books*, 1977.

5. Cox, D. R. and Smith, W. L., *Queues*, London: Methuen & Co., Ltd., 1961.

6. Sherbrooke, C., "METRIC: A Multi-Echelon Technique for Recoverable Item Control," Rand Report RM-5078-PR, November 1966.

7. Hadley, G. and Whitin, T. M., *Analysis of Inventory Systems*, Prentice-Hall, Inc., 1963.

8. Fishman, George S., *Principles of Discrete Event Simulation*, Wiley Inter-science 1978.

9. *Air Transport Association Inventory Planning Manual*, revised June 1, 1976.

10. Wilson, R. H., "A Scientific Routine for Stock Control," *Harvard Business Review XIII*, 1934-1935.

APPENDIX A

FAULT-TOLERANT FLIGHT CONTROL SYSTEMS

CONTENTS

FIGURES

TABLES

A.0 FTFCS DESIGN CONCEPTS

If the CBOM is to be a useful tool, it must be capable of modeling any reasonable FTFCS design. In order to provide this capability, the model builder requires a thorough understanding of fault-tolerant systems in general and fault-tolerant flight control systems in particular.

Appendix A presents the information that was collected to establish the levels of abstraction and detail required to provide fault-tolerant flight control system (FTFCS) descriptions as inputs to the Cost and Benefit Optimization Model (CBOM). The goal was to provide to the CBOM a sufficient set of input parameters so that the important subtle features of realistic FTFCS designs could be modeled while avoiding excessive detail that would make the model unworkable. The accomplishment of this goal required an understanding of both fault-tolerant techniques and the commercial aircraft flight control environment.

Current designs and brass boards do exist for the core of an FTFCS; this core is envisioned to be an extremely survivable, centrally located FTFCS computer capable of performing such life critical functions as active controls, total fly-by-wire, and total system management. There are two candidate FTFCS central computer architectures. The Software Implemented Fault-Tolerant (SIFT) system, designed by SRI International, is being built as a NASA engineering prototype by the Bendix Corporation. The Fault-Tolerant Multiprocessor (FTMP), designed by Charles Stark Draper Laboratories, is being built for NASA flight tests by Collins Radio, and both are engineering prototypes.

A.1 FCS DESIGN OVERVIEW

This section presents a brief overview of current practices and trends in the design and implementation of commercial aircraft flight control systems. The information is drawn from a survey of existing and proposed flight control and flight management systems for commercial aircraft currently under production by the Boeing Commercial Airplane Company, including the 727, 747, 757, and 767 Boeing jetliners.

Fault-tolerant central flight control computers, currently under consideration as the computational core of an FTFCS, are designed to support the implementation of an integrated flight control system that has been designed for a system-wide top-down approach. The goal of a top-down approach is to produce a more efficient flight control system devoid of unnecessary duplication of functions that is, in general, better understood and verified. Contrast this ideal to the automatic flight control system shown in Figure A-1 in which major control functions are not only functionally separate but are usually implemented in separate control computers.

The transition from current FCS technology to commercial aircraft using fly-by-wire and a complete FTFCS will, of course, take place in careful, studied steps. One of these will be the move from flight control hardware, which is generally analog, to flight control hardware, which is generally digital. This particular transition is taking place in the development and upgrade of aircraft currently under production at the Boeing Commercial Airplane Company. The result thus far has been the production of hybrid flight control systems in which the problem of interfacing a vast array of dissimilar analog and digital devices has become a major area of FCS design.

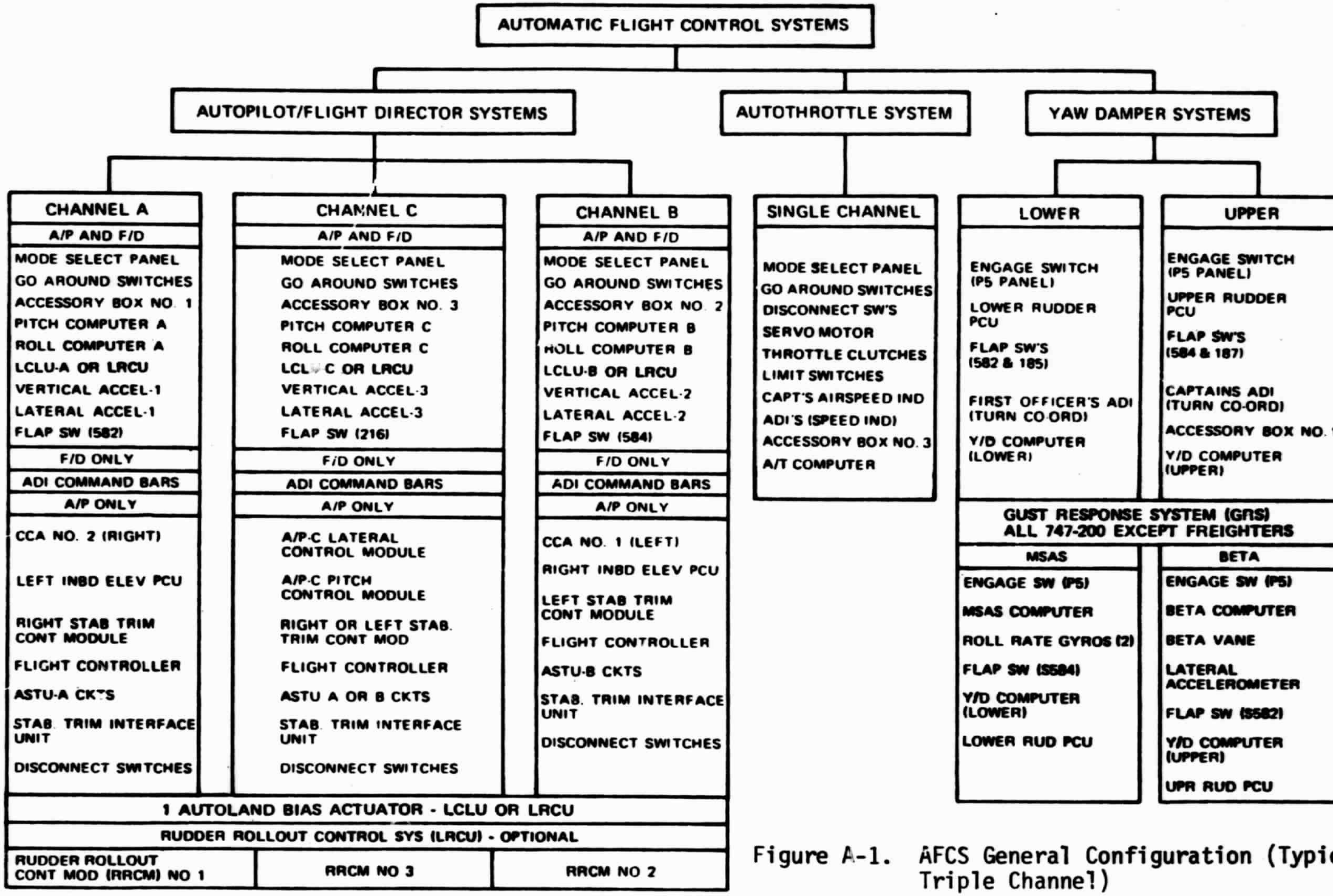


Figure A-1. AFCS General Configuration (Typical Triple Channel)

Figure A-2 shows how several major components of the Boeing 767 flight management system communicate via an ARINC 429 standard digital data bus.

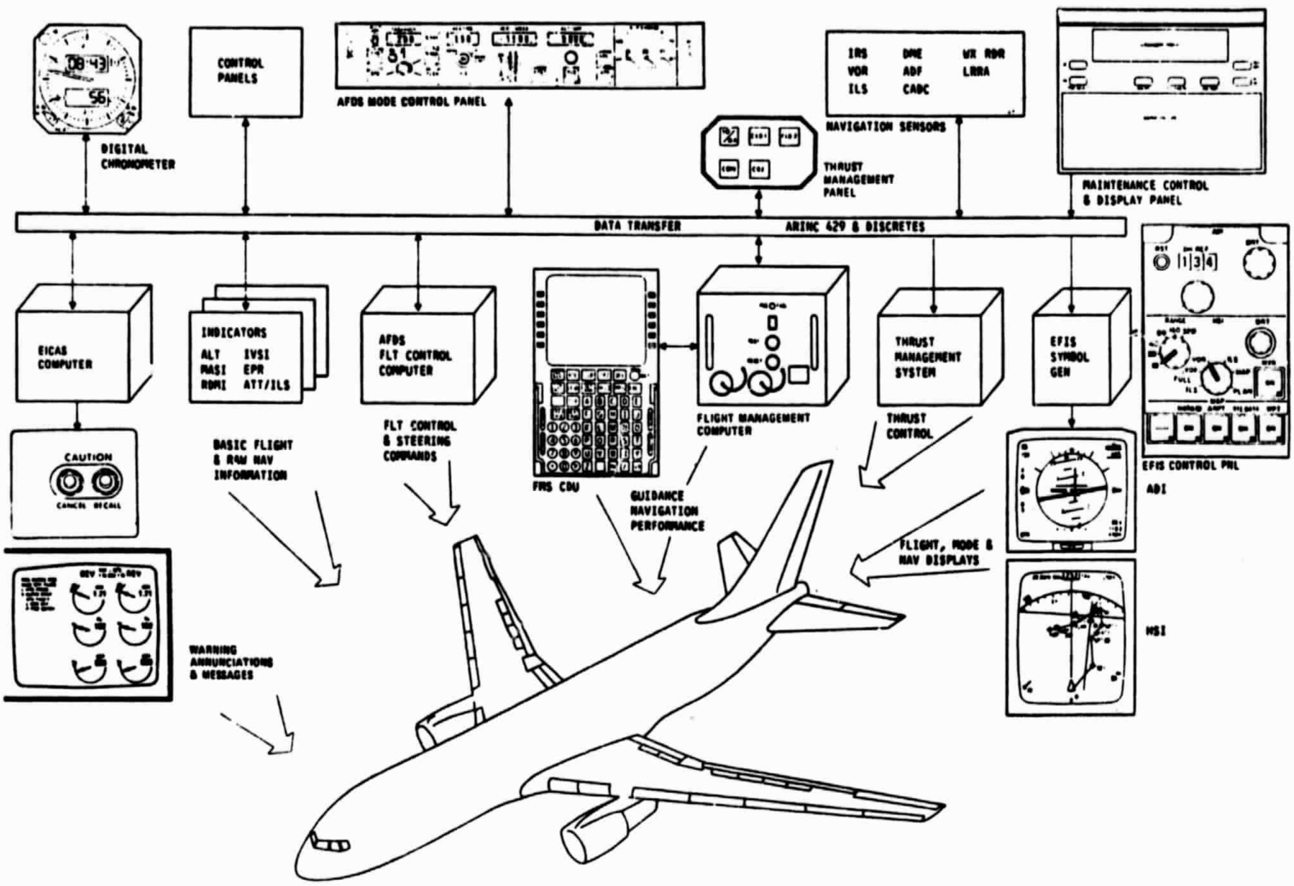


Figure A-2. 767 Flight Management System

A.2 FTFCS PHYSICAL CHARACTERISTICS

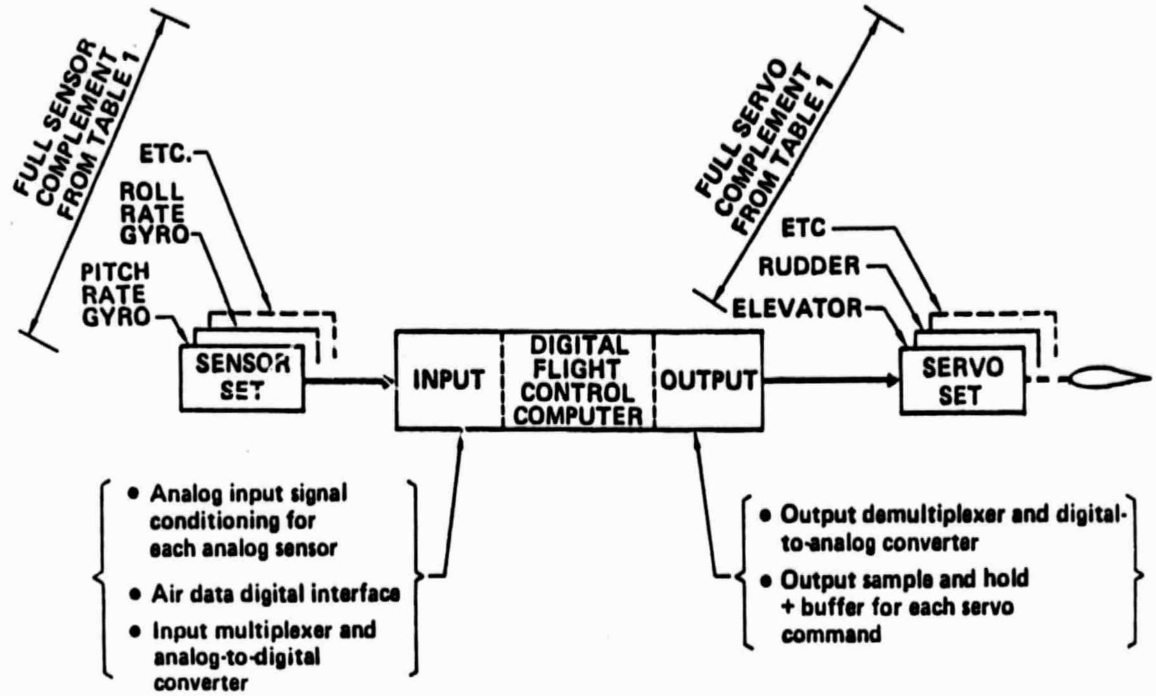
A survey of proposed and existing fault-tolerant systems suitable for flight control applications revealed nearly as many different (and often opposing) design philosophies as systems examined; however, all of these apparently diverse systems make use of a number of easily defined fault-tolerant architectural features and techniques. The repeated occurrence of this array of fault-tolerant mechanisms allows a systematic and hierarchial approach to the definition of a fault-tolerant architecture, especially in the case of less sophisticated systems where the implementation details of these basic mechanisms are not clouded by overall system complexity.

A.2.1 Primitive FCS Architectures

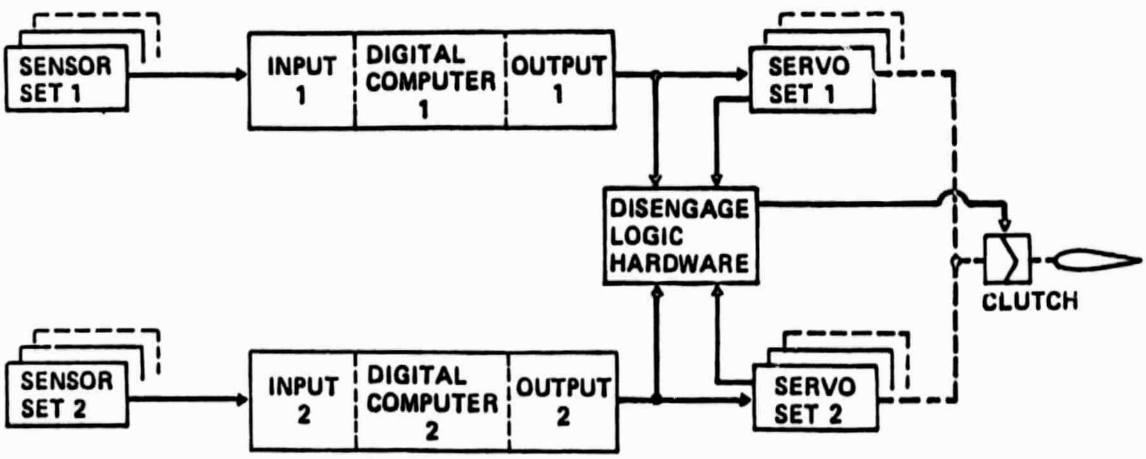
This section examines several examples of primitive attempts at implementing fault-tolerant concepts in flight control applications. None of the systems meets either the reliability or the functional (i.e., computational) requirements of advanced airplane guidance and control systems. However, the systems are useful for providing examples of many of the basic fault-tolerant mechanisms used by fault-tolerant systems at all levels of sophistication.

During the past decade, a great deal of effort has been directed toward the design and implementation of reliable digital systems in military and civilian flight control applications. Rice and McCorkel (refs. 1 and 2) provide a good overview of what might be called the primitive approach to fault-tolerant digital flight control systems. Figure A-3 provides illustrations of some of the systems considered in this overview.

ORIGINAL PAGE IS
OF POOR QUALITY



(a) Single-channel (simplex) system

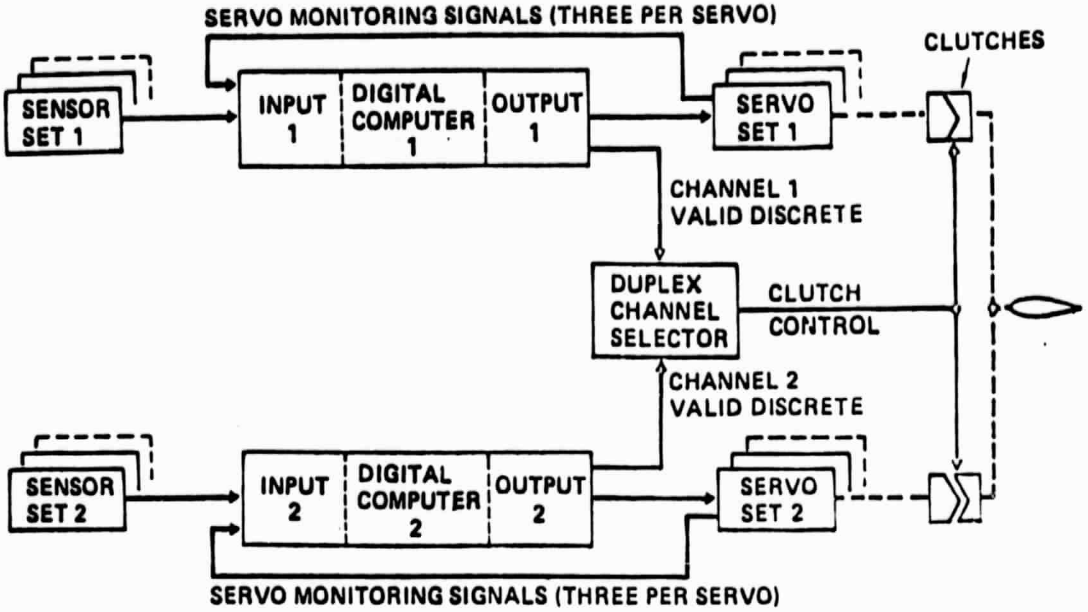


Note; Each above channel same as simplex channel

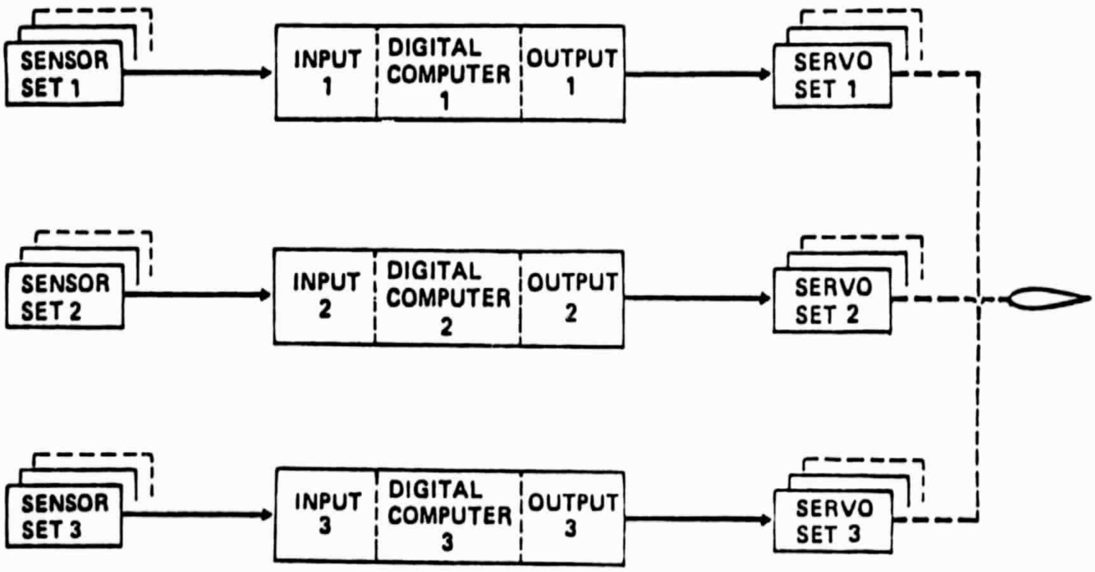
(b) Conventional dual-channel fault-passive system

Figure A-3. Overview of Fault-Tolerant Digital Control Systems

ORIGINAL PAGE IS
OF POOR QUALITY



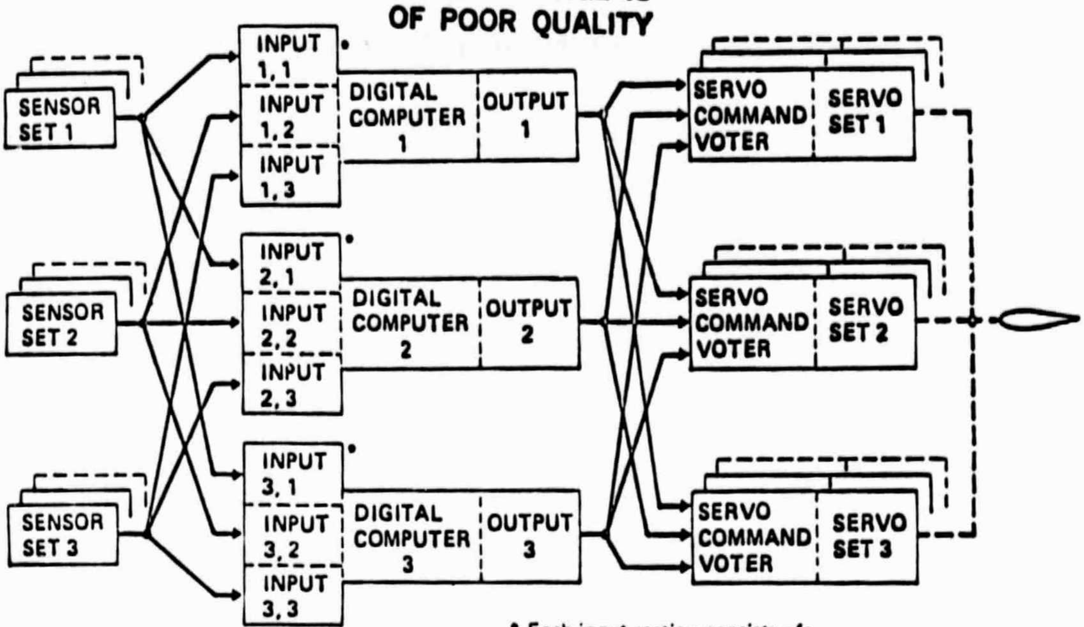
(c) Dual-channel system with channel selection redundancy management



(d) Triplex voted system--servo force voting only

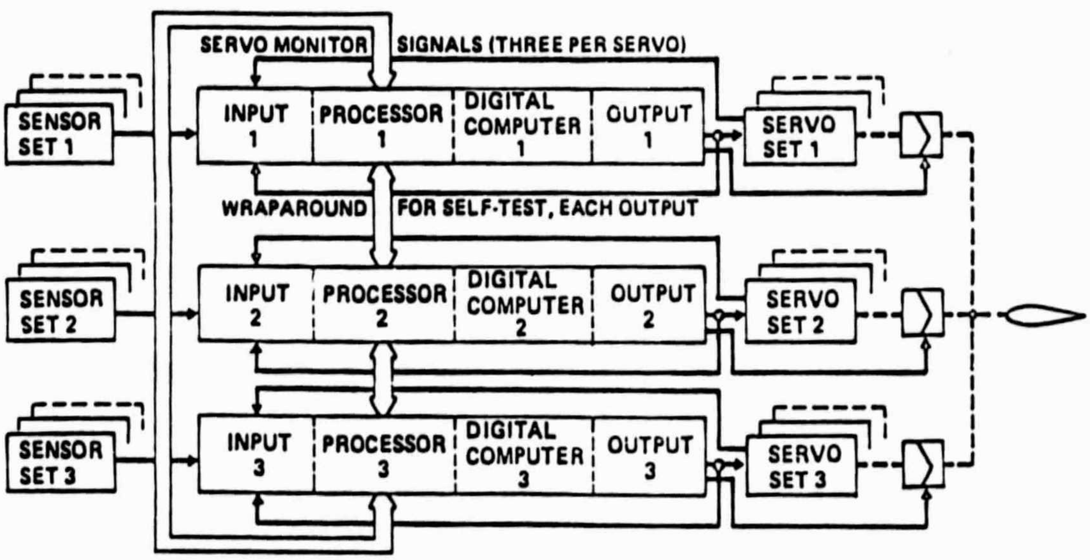
Figure A-3. Overview of Fault-Tolerant Digital Control Systems (Continued)

ORIGINAL PAGE IS
OF POOR QUALITY



- Each input section consists of:
  - A set of analog input signal conditioning channels
  - An air data digital interface
  - An input multiplexer and analog-to-digital converter

(e) Triplex voted system--servo force voting, servo command voting, and cross-strapped sensor signal voting

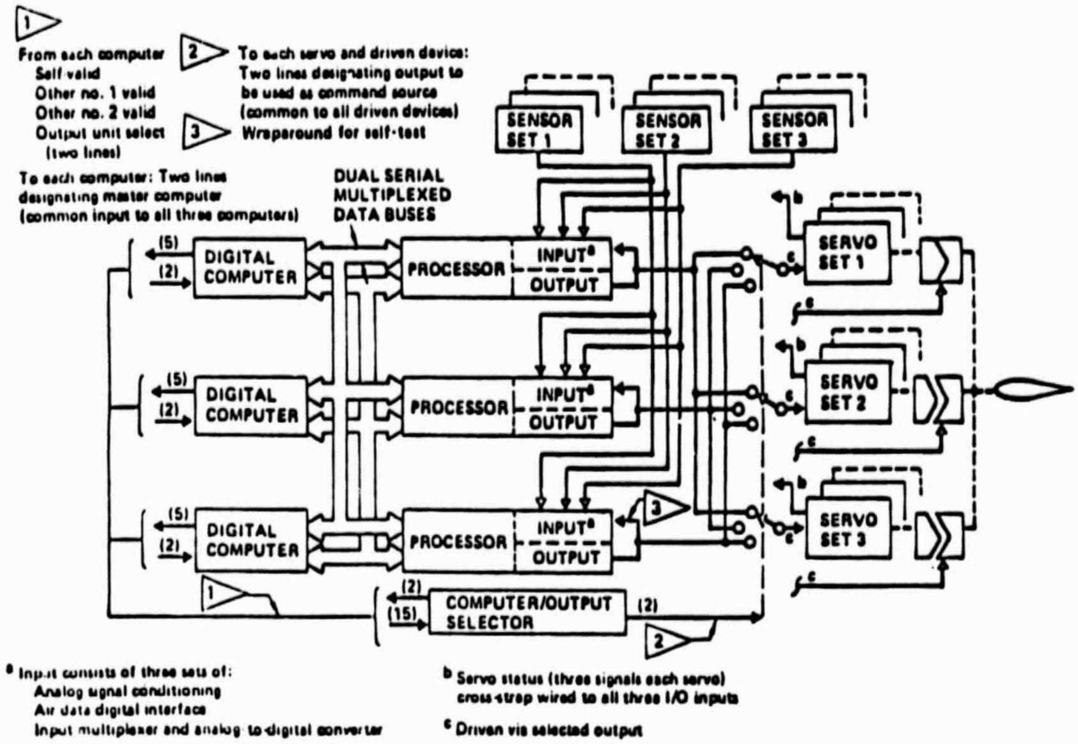


- Input, processor, computer, and output of a channel must be good for that channel's servo to be engaged
- Input and processor must be good to send sensor data to another channel
- Sensor data redundancy managed for successful degradation to simplex for each sensor function
- Channel failures sustainable to simplex level of operation

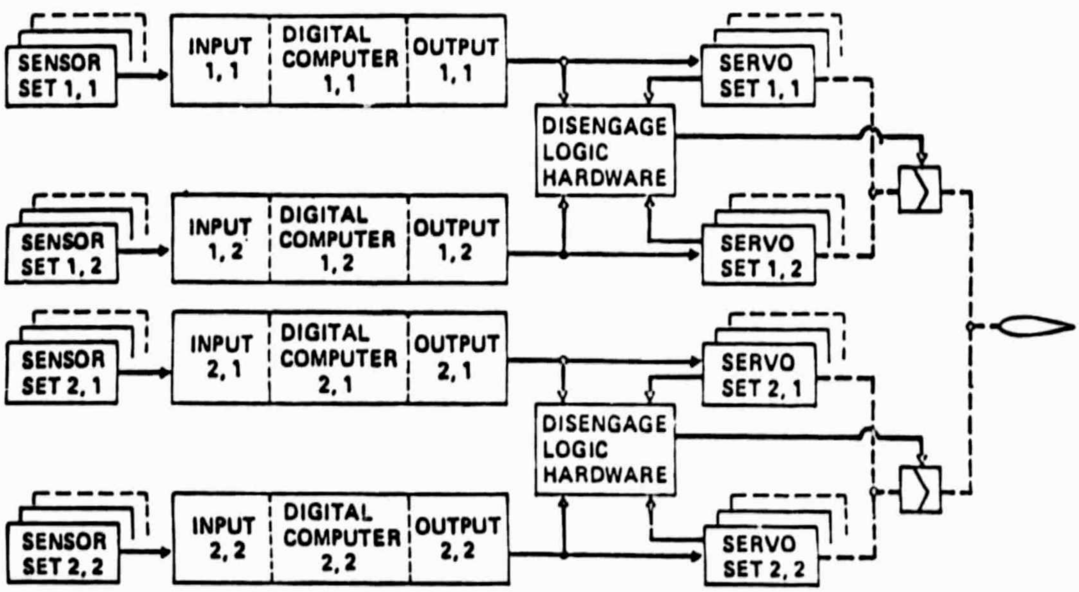
(f) Triplex ARCS concept

Figure A-3. Overview of Fault-Tolerant Digital Control Systems (Continued)

**ORIGINAL PAGE IS
OF POOR QUALITY**



(g) Triplex system with redundancy management and interunit selection



(h) Dual dual system

Figure A-3. Overview of Fault-Tolerant Digital Control Systems (Concluded)

Figure A-4 (ref. 1) compares the probability of failure, as a function of mission time, of each system illustrated in Figure A-3. Note that none of these systems meets the reliabilty requirements for commercial flight crucial systems.

Common to each of the architectures is the inherent single stream approach (as opposed to a multiprocessor approach) and the final reliance on some fail-safe backup system. Both of these design constraints are in direct conflict with the functional and reliability requirements of an FTFCS and provide the justification for labeling such flight control architectures as primitive. It is possible that a leap in digital technology could result in highly reliable and computationally powerful system components so that even a simplex, single-stream flight control system could satisfy the reliability and functional requirements of an FTFCS. However, such dramatic technological advances are not likely in the 1990's time frame with current evolutionary progress. Neither single-stream flight control systems nor flight control systems that rely mainly on fault avoidance over fault tolerance will be considered as economically feasible candidates for performing flight crucial functions.

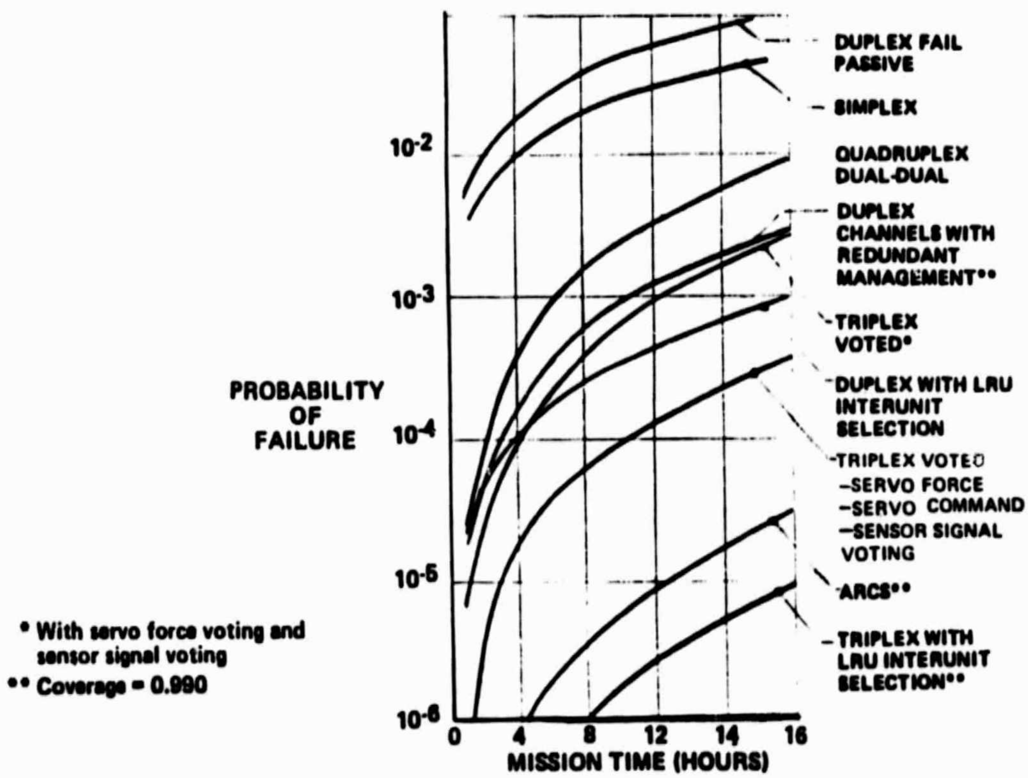


Figure A-4. System Comparison

A.2.1.1 Key Features of Primitive FCS Architectures

Given that a flight control system falls under the category of primitive fault-tolerant architecture, the first step in the definition of the system is to determine the gross level of redundancy. This step can be as simple as differentiating between simplex, duplex, triplex, quadruplex, and so on. The existence of hybrid redundancy schemes such as the dual dual system (ref. 1, fig. A-3h) complicates this task somewhat, but a careful examination of such systems usually reveals that they are a refinement of a general level of redundancy. For instance, in the case of the dual dual system, two main channels are employed, each of which is itself a duplex channel. It is clear that for either main channel to be nonfailed the entire duplex channel set comprising that main channel must be nonfailed. Thus, the gross redundancy level of the system is still duplex.

The next step in defining a primitive fault-tolerant system is to detemine the degree and implementation of fault-tolerant mechanisms such as channel selection, voting, and reconfiguration schemes. An FTFCS employing any degree of physical component redundancy must have some means of detection and isolation of failed components. This task may be accomplished by various combinations of several easily identifiable methods that are tabulated and explained below:

Physical Comparators—The most simple method of fault detection uses a device that compares the outputs of two or more system components and, upon discovery of any disagreement, initiates some sort of fail-passive procedure—often a function disengagement. The duplex system shown in Figure A-3b employs such a device to compare control computer outputs and disengage the actuator clutch when a disagreement occurs. The dual dual system of Figure A-3h is a somewhat more complex version of this scheme and is used where hardover or oscillatory failure must be avoided.

Physical Channel Selector—The next level of fault detection and handling requires self-test and servo monitoring by each channel's control computer to detect failures. Detected failures are reported to a physical channel selector device that contains some means, such as clutch disengagement logic, to switch channels in the presence of a single failure or, if a second failure occurs, to disengage both channels. The duplex system shown in Figure A-3c uses self-test discretes from the control computers to select or disengage the online controlling channel. If both channels report invalid operation, a fail-passive fuction disengagement takes place.

Voting—The next level of sophistication found in a primitive fault-tolerant flight control system is the employment of physical and logical voting of signal paths and computational results. Three main types of voter implementation will be distinguished: actuator force voting, analog electronic voter circuits, and software implemented logical voting schemes. An example of a system employing only physical voting is the triplex system with acutator force voting shown in Figure A-3d. The servo command voters in the triplex system shown in Figure A-3e introduce electronic voter circuits; this system also employs direct wiring of cross-strapped sensor inputs, allowing logical voting of sensor data in each channel's control computer.

A common feature of the system shown in Figures A-3d and A-3e is that all signals required for voting use dedicated hard-wired signal paths directly to the physical voting device or logical voting plane. The systems shown in Figure A-3f and A-3g use a general purpose data bus for interchannel communication. This feature allows logical voting operations to occur in each channel and permits exchange of signal values and computational results rather than relying on dedicated hard-wired signal paths for voting operations. Interchannel communication represents a significant departure from the hard-wired approach to multichannel flight control applications.

Reconfiguration—None of the primitive flight control systems examined made use of full reconfiguration. Some do allow partial reconfiguration capabilities. For example, the triplex ARCS concept (fig. A-3f) disengages one channel's servos upon detection of failure in that channel but, depending on the source of the failure, may still allow the failed channel's I/O processor to supply sensor data to the remaining two nonfailed channels. The triplex system of Figure A-3g extends the features of the ARCS concept to include dual redundant interprocessor data busses and more complex servo input selector device. The important feature in both cases is the ability to maintain use of nonfailed components in a flight control channel in which a failure has occurred. This is possible because various input, command, and computational data may take any of several paths through the different channels of a system via the interchannel digital data busses. Of course, the usefulness of this capability hinges on the failure recognition and handling capabilities of the executive software as well as the ingeniousness of design and the reliability of any signal selection devices or switching circuits.

The common fault-tolerant mechanisms found in the survey of primitive FCS architectures are:

- Redundant channels (figs. A-3b through A-3h)
- Physical comparators (fig. A-3b)
- Physical channel selectors (fig. A-3c)

- Voters

    - Servo actuator forced voters (fig. A-3d)
    - Electronic vote circuits (fig. A-3e)
    - Logical voting planes (fig. A-3f)

- Reconfiguration mechanisms

    - Component degradation and alternate signal paths (fig. A-3f)
    - Redundant data busses (fig. A-3g)
    - Output-select switching circuits (fig. A-3g)

A.2.1.2 Sensor/Actuator Redundancy Management and Signal Communication in Primitive FCS Architectures

It is a common feature of all of the primitive FCS architectures examined that little or no attempt is made to advance the state of the art of redundancy management at the sensor or actuator planes. With the exception of the triplex FCS of Figure A-3e that combines servo command voter circuitry with the servo actuator electronics, all sensor and actuator redundancy management functions are performed by the control computers and their associated input and output and switching circuity. Further, the conventional practice of using shipboard wiring for sensor and actuator analog data communication to and from the control computer section is universally followed; analog signal conditioning and digital-to-analog and analog-to-digital conversion functions are the responsibility of the input and output circuitry associated with the digital control computers.

The specific circuitry employed in the design of the input and output hardware of the primitive FCS designs discussed so far is device and environment dependent. A description of the components used for this task will change with each new set of sensors or actuators and with each new shipboard wiring environment. What can be said about these components is that their complexity and corresponding size will be directly proportional to the number of different devices with which they must communicate. This effect is illustrated in Figure A-3e where direct wired cross-strapped sensor inputs are supplied to each channel of a triplex FCS. An additional effect of this approach is the attendant penalty in aircraft wiring weight and input connector requirements (ref. 1). Systems such as those of Figures A-3f and A-3g avoid the heavy wiring and signal handling circuitry duplication penalties of the cross-strapped sensor input approach, but the corresponding increase in sensor redundancy management complexity requires their use of separate input and output processors.

An alternative approach to that taken by any of the primitive FCS designs surveyed for this report is to remove the burdens of signal conditioning, signal conversion, and sensor and actuator redundancy management tasks from the computing section of an FCS. The benefits gained include a decrease in shipboard wiring requirements, a decrease in input and output hardware and switching circuitry complexity, and an increase in system flexibility. Two primary requirements of this approach are the development of digital sensor and actuators electronics and the implementation of some kind of fault-tolerant shipboard digital data bus scheme with sensor and actuator redundancy management capabilities at the site of remote teminals. This subject is explored further in Section A.2.2.

In summary, the crucial and critical flight control functions are separately implemented in contemporary systems and, in all cases, the capability exists of ultimate reversion to control by the pilot or by some trusted backup device. This approach certainly has historical precedence and, while it does little to advance the state of the art of fault-tolerant techniques, it does provide the comforting final reliance on techniques of known structure and reliability. The advent of inherently unstable, fly-by-wire, active control commercial aircraft will radically

change the concepts of the pilot's function in an FCS and the use of backup systems in general. The following summary of primitive FCS architectures emphasizes those features that tend to make such systems unsuitable FTFCS candidates:

- **Component Redundancy and Reconfiguration**—The general approach seen in primitive fault-tolerant designs involves the use of single-stream computational section employing successively higher numbers of redundant channels as reliability requirements are increased. The major flaw of this approach is that a limit is reached beyond which a simple increase in redundant channels without a corresponding increase in the intelligence of reconfiguration mechanisms can actually result in a decrease in system reliability (ref 3). Some primitive fault-tolerant architectures do introduce a certain amount of reconfiguration capabilities (for example the ARCS system of fig. A-3f). None had the capabilities to create and dismantle redundant processing streams or to manage spare component pools.

- **Digital Data Busses**—Brick wall FCS architectures such as the triplex system of Figure A-3d have no requirement for digital interchannel communication. General purpose digital data bussing schemes begin to appear in those primitive fault-tolerant systems that depart for the brick wall approach in order to support sensor redundancy management capabilities and limited component reconfiguration (fig. A3-f). A fault-tolerant system-wide digital bus structure is a basic requirement of a system employing general component reconfigurability.

- **Self-Test**—Some of the primitive fault-tolerant systems examined in this section (e.g., figs. A-3c and A-3g) rely on certain components for detecting and reporting their own failures for correct system operation. This is a nonsatisfactory practice in the case where a failure mode resulted in a failed unit reporting itself as a nonfailed component.

- **Channel and Signal Selectors**—The systems that use discrete components are simply not suitable as choices for an FTFCS. Such a system can never be more reliable as a whole than the reliability of the individual fault monitors or unit selectors themselves. When the burden of fault detection, reporting, and isolation is shared among several main system components, none of which has individual precedence or priority over another, the system is fully fault tolerant. Examples of this technique are presented later in this report.

- **Sensor and Actuator Redundancy Management**—The total reliabilty of a given FTFCS is not only a function of the reliability of the core computer but also of the reliability of busses, sensors, and actuators ultimately supplying or receiving crucial or critical flight data. Given that reliable sensors, actuators, and shipboard data busses will result from the use of redundancy, there then will exist the need for the management of a fairly complex network of physically and functionally distributed components. This task probably can best be accomplished by the introduction of digital data busses, digital data bus remote terminals, and digital actuator and sensor electronics that are themselves fault tolerant. None of these issues have been addressed by the primitive fault-tolerant systems examined. Any management of redundant data paths or sources has been left as an additional burden on the central computational section.

- **Signal Conditioning, Input and Output**—Conditioning and conversion of sensor and actuator data, whether analog or digital, in the primitive fault-tolerant systems examined thus far takes place in the input and output sections of the control computers proper. This not only results in complex input and output sections with each section containing circuitry to support all devices it might have to communicate with, but it also results in the structure of a given FCS being highly environment dependent. Unless all commercial aircraft can be counted on to supply the same or similar sets of sensor, actuator, data bus, and signal line requirements to the control computers, then the input and output sections of the control computers will be highly nonstandard from one installation to the next. Neither this subject nor the sensor and actuator redundancy management discussed above have been sufficiently addressed by either the primitive fault-tolerant architectures examined in this section or the fully fault-tolerant flight control computers examined in later sections. An example of a scheme by which these problems may be handled, however, is presented later.

The pressure of rising fuel costs and the expanding technology of advanced digital systems will combine to make integrated active digital control systems inevitable but, in the meantime, the foundations for the eventual use of such systems will consist mainly of discrete control functions implemented with increasing degrees of sophistication of fault-tolerant techniques. It is likely that primitive fault-tolerant designs, such as those reviewed, will provide the technological bridge between non-fault-tolerant FCS designs and the implementation of modern, sophisticated, fully integrated FTFCS designs examined in the following section.

A.2.2 Modern FTFCS Architectures

This section examines more sophisticated fault-tolerant systems that are likely to meet the reliability and functional requirements of an FTFCS. Primary attention is given to FTFCS central computers, especially SIFT and FTMP. External devices such as sensors, actuators, data busses, and remote terminals are also discussed but, as no fully integrated FTFCS architecture exists for these devices, the examined systems will be hypothetical.

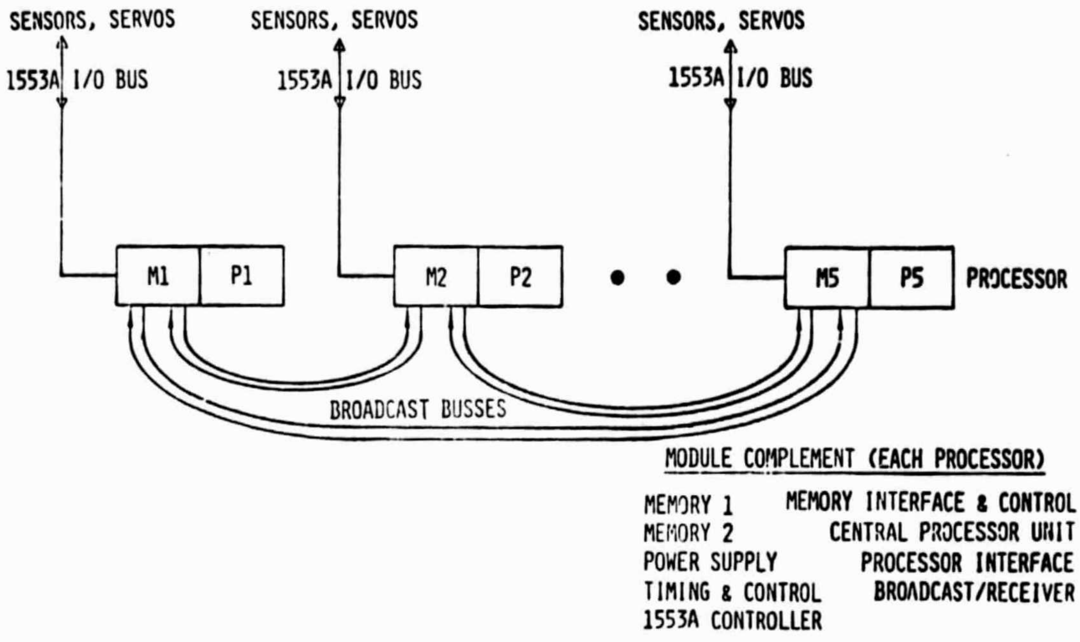
Each of the discussed FTFCS architectures can be considered as reasonable candidates for modeling by the CBOM. Comparing the central control computer section of any of the primitive FCS architectures to the highly replicated, highly reconfigurable, functionally degradable multiprocessor approach as taken by either SIFT or FTMP (refs. 4 and 5) reveals an important level of sophistication in the defining of FTFCS types. What is different about an FTFCS built around a SIFT or an FTMP is the reconfigurability and anonymity of all major system components. Thus, in an FTMP, for example, processor triads making up single processing channels or streams of a multiprocessor can be created and dismantled according to the dynamic needs of the system with no need for external authority or watchdog devices. This feature of general reconfigurability will be used to differentiate between a primitive and a modern fault-tolerant architecture.

A.2.2.1 SIFT Architecture

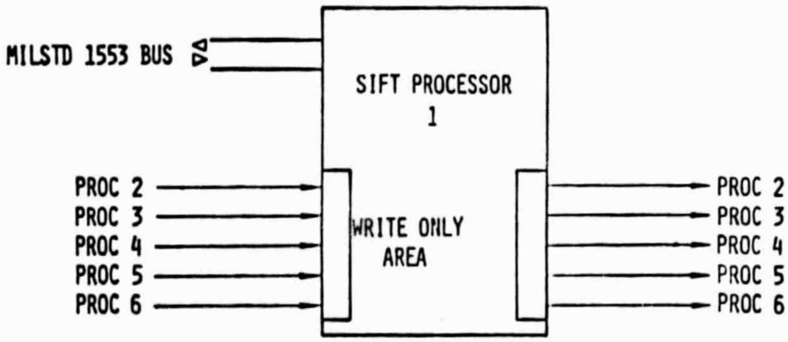
As the name implies, SIFT is a fault-tolerant system in which reliabilty results from software techniques rather than through hardware fault-tolerance and fault-

avoidance mechanisms (ref. 4). That is, SIFT achieves fault tolerance through its task allocation strategies and through voting mechanisms and error isolation mechanisms built into the operating system.

The hardware architecture to support fault-tolerant operations in SIFT is remarkably simple. SIFT consists of up to eight processors connected to each other by a broadcast interface (fig. A-5a). Each processor has its own local memory with a copy of every SIFT task. Each processor communicates with external sensors and actuators via a MIL-STD-1553A serial data interface. Figure A-5b depicts the processor interface for one SIFT module.



(a) Redundancy view



(b) Processor communication via broadcast bus

Figure A-5. FTFCS Central Computer--SIFT

There are no built-in test devices, error correcting/detecting busses, component isolation devices, or special equipment to enhance reliability or detect malfunctions.

The current SIFT processor is a stock Bendix BDX 930 designed primarily for avionic applications. The main memory contains 30K, 16-bit words and both system and flight applications programs. A 1K, 16-bit scratch pad or data file is used to store the temporary results produced by the processor's tools. A 1K, 16-bit transaction file is used to control the configuration and destinations of task outputs. The external bus is a MIL-STD-1553A serial half-duplex link. Each 1553A controller can support up to 31 remote terminals with associated actuators and sensors. The broadcast interface is simply a write-only area in every processor that any given processor can access. The destination write areas for each piece of information produced by SIFT is stored in the transaction file (ref. 6). Each processor, memory, and 1533 controller occupies a standard ½ ATR short LRU. The fundamental physical characteristics of the SIFT processor module are:

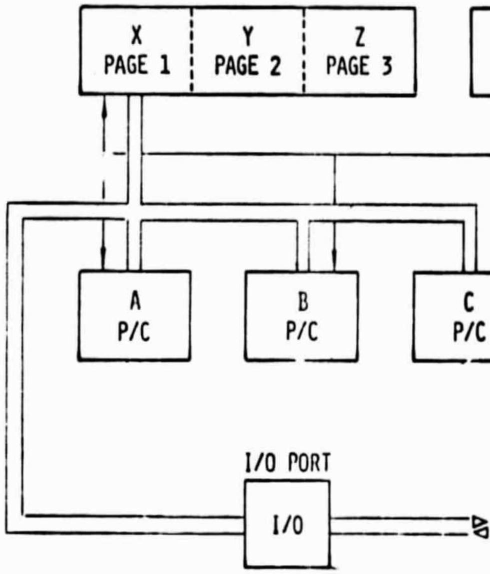
- LRU size: 4.88 x 7.7 x 12.52 in
- Environment: cabin conditioned
- Power requirements: 28V, 1W battery backup
- Estimated MTBF: 6500 hr
- Interconnections: broadcast, write only
  interface to all five other processors
- Cost: $27 000 (1979 dollars)
- Throughput: 500K ops Gibson mix-raw
- Inputs: (MIL-STD-1553 external data bus/1 MHz 32 ports)
- Outputs: (MIL-STD-1553 external data bus/1 MHz 32 ports)
- Minimum number of this component required for successful operations: four
- Standard number of this component available: six
- Maximum number: eight

A.2.2.2 FTMP Architecture

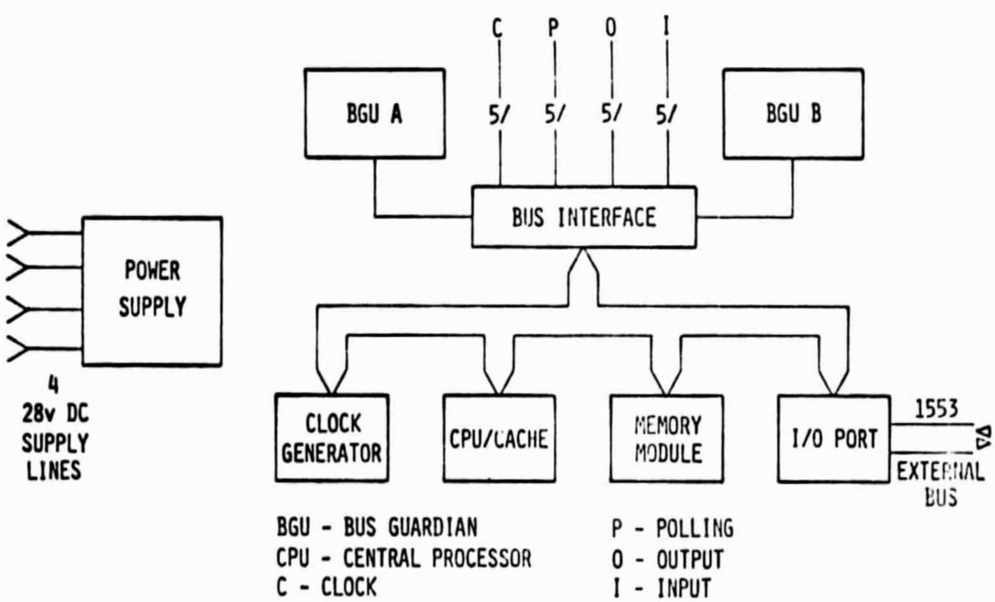
The FTMP is a fault-tolerant system in which reliability results from fault-tolerance and fault-avoidance mechanisms along with software implemented component reconfiguration mechanisms built into the operating system (ref. 5).

The FTMP operates as a highly reliable three-stream multiprocessor consisting of an independent processor and cache-memory for each channel. All three units communicate, via serial bus lines, with a single three page mass memory and

several task dedicated I/O ports. This multiprocessor viewpoint is shown in Figure A-6a. The fault tolerance of FTMP comes from the fact that triple modular redundancy (TMR) is employed for each processor channel, each memory page, each data line, and each module's incoming clock signal. Hardware bit-by-bit voting is performed on all data transfers and all single errors are masked by taking a majority value (two out of three vote). An executive program periodically searches the system for set error-latch registers, reconfigures the systems (by reassigning bus-module associations) to pinpoint disruptive modules, and takes failed units off line, replacing them with active spares.



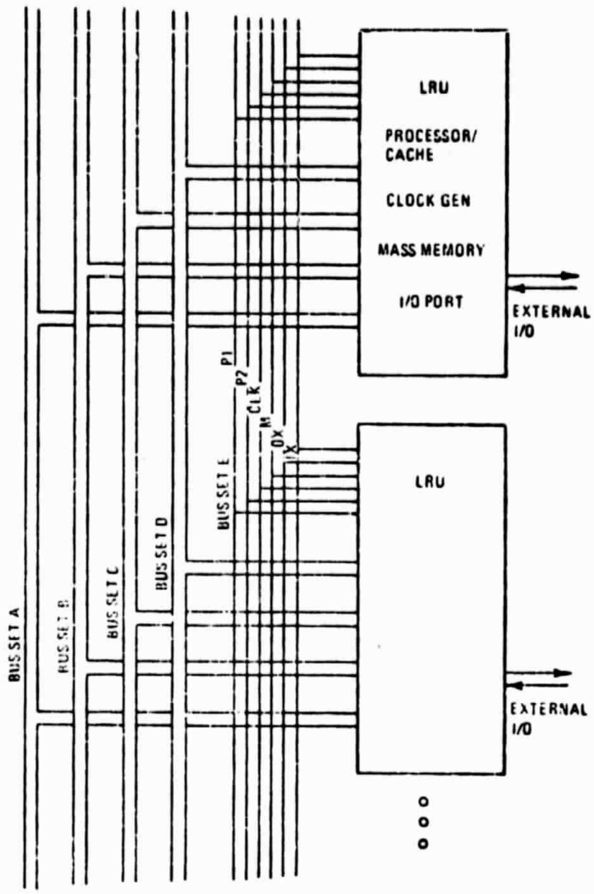
(a) Multiprocessor view



(b) LRU main components

Figure A-6. FTFCS Central Computer--FTMP

ORIGINAL PAGE IS
OF POOR QUALITY



(c) Bus system

Figure A-6. FTFCS Control Computer--FTMP
(Concluded)

With the exception of bus lines, all components in the FTMP system are contained in 10 identical LRUs. Each LRU contains the following items:

- One CPU/cache
- One 16K main memory module
- One clock generator
- One power supply
- One I/O port
- Bus interfaces
- Two bus controllers (BGU)

Of these, the CPU/cache, the main memory module, the clock, and the I/O port are fully reconfigurable. The LRU itself is not a reconfigurable part. Figure A-6b shows the FTMP LRU main components.

ORIGINAL PAGE IS
OF POOR QUALITY

Table A-1 contains the definitions for the reconfigurable components of an FTMP.

All components are contained within the data shown in the LRU column plus additional cards that are unique to the LRU.

Each LRU contains its own power supply, which like the BGUs and the bus interface, is nonreconfigurable. All components within the LRU are supplied 5V dc by this power supply. The power supply itself draws 28V dc from a quadruple redundant system-wide main power source.

In addition to those components contained in the FTMP LRU, there are a total of 20 back-plane-mounted bus lines in an FTMP. "The bus system (shown in fig A-6c) is quintuple redundant. Each bus has lines dedicated to processor transmission (the two P bus lines); memory module transmission, (the M line); clock generator transmission, (the CLK line); and I/IO transmissions, (the IX and OX lines). Subsets of three of the five busses are assigned to carry processor and memory triad data. A subset of four of the five is used to carry clock generator transmissions. A single bus of the five is used to carry I/O port transmissions." (This description of the FTMP bus structure is taken from ref. 5.)

TABLE A-1. FTMP Module and LRU Characteristics

	PROCESSOR CACHE	MEMORY	CLOCK	BUS LINE	IO PORT	LRU
Size (cards)	8-½ ATR	2-½ ATR	1-½ ATR	N/A	3-½ ATR	21-½ ATR
Weight (kg)	N/A	N/A	N/A	N/A	N/A	18.14
Power	5V dc	5V dc	5V dc	28V dc	5V dc	28V dc 150W
Inter-connections	Clocks memories IO ports	Processor	All other	Each LRU modules	Bus lines processor caches sensors actuators	Other LRUs
MTBF (hr)	20 000	20 000	30 000	N/A	30 000	2000
Cost	N/A	N/A	N/A	N/A	N/A	$35 000 (1979 $s)
Throughput	500K ops Gibson Mix	N/A	N/A	N/A	N/A	N/A
Typical number	10	10	10	20	10	10
Minimum	2	2	4	5 P lines 3 O lines 3 I lines 4 C lines	1	4

A.2.2.3 Sensor and Actuator Redundancy Management and Digital Data Communication in Modern FTFCS Architectures

Given the number of very diverse aircraft environments in which a successful FTFCS central computer might have to function, a modular design approach is most reasonable. It would be highly undesirable for the architecture of an FTFCS control computer to depend on the particular array of sensors and actuators required on a particular aircraft as is the case for most previous attempts at implementing fault-tolerant techniques in a flight control system. While it is not the intention of this report to solve the many problems of the integration of external FTFCS devices to a fault-tolerant central control computer, it is a primary intent that when complete modern FTFCS architectures do emerge, the CBOM be capable of handling them. The following material presents a best guess description of how a fully integrated FTFCS architecture might appear based on current commercial FCS practices.

Figure A-7 shows a high level block diagram of a possible fully integrated FTFCS architecture. The key features of this FTFCS architecture include a fault-tolerant central computer, a fault-tolerant external data bus, a number of fault-tolerant remote terminals for the fault-tolerant external data bus, and an array of redundant sensor and actuator sets. For a more detailed examination of this FTFCS architecture, assume that the fault-tolerant central computer is an FTMP and that the fault-tolerant external data bus is a set of MIL-STD-1553A data busses, one for each FTMP LRU.

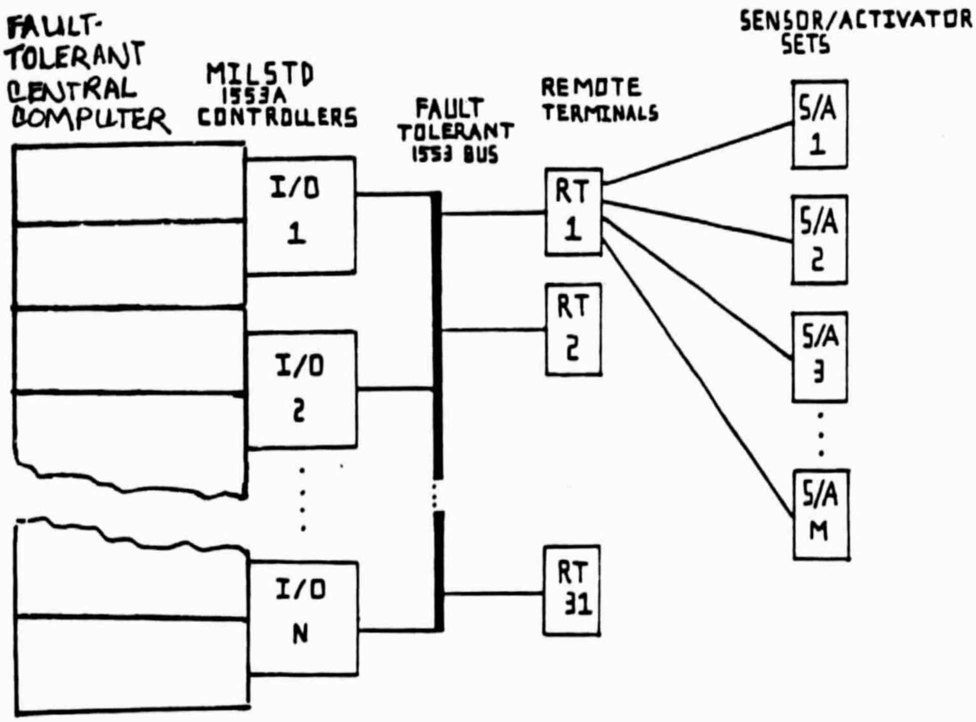


Figure A-7. Possible Integrated FTFCS

The only component in this proposed FTFCS architecture for which no existing hardware or designs exist is the component designated as a fault-tolerant remote terminal. The concept of this device is that it will perform the functions that follow.

Device-Set and Bus Line Multiplexing—Most serial data bus architectures impose a limit to the number of remote terminals allowed in the network. In the case of a MIL-STD-1553A bus, this number is 31; also, it is highly desirable that each FTMP I/O port be able to communicate with every remote terminal that is connected to devices performing critical or crucial functions, the number of which is likely to be far in excess of 31. Therefore, the proposed fault-tolerant remote terminal must be a device that is capable of interfacing multiple MIL-STD-1553A bus lines to multiple sensor and actuator sets.

Signal Conditioning and Conversion—Sensors and actuators are typically devices that send and receive analog signals while the rest of the proposed FTFCS components communicate via digital data busses. The point of interface between these analog devices and the digital central computer is the proposed fault-tolerant remote terminal. In order to minimize the complexity of shipboard wiring, the most desirable location of these remote terminals is at or near the site of sensor and actuator sites. Not coincidentally, this is also the optimal location for the electronic conversion, analog signal amplification, and various other signal conditioning functions required by the sensor and actuator sets. While it is not desirable for this specialized and device-dependent circuitry to be an integral part of what should be a general purpose remote terminal, it is likely that initial implementations of this type would require at least special interface circuity to be a part of each specific remote terminal design and application.

Advances in digital technology will probably make possible standard digital interfaces to these analog processes, allowing the proposed fault-tolerant remote terminal to be truly general purpose. In fact, VLSI technology may allow the integration of the functions not the design of the sensor or actuator component proper, making these inherently analog devices appear as digital devices with perhaps a simple, standard digital interface provided.

Sensor/Actuator Redundancy Management—Because of the limited number of devices that can be attached to a 1553A controller and because it is desirable to minimize the amount of multiplexing capabilities required of a given remote terminal, it is unlikely that the data or signals required by or supplied from each individual sensor and actuator in a given FTFCS will be passed between the fault-tolerant central computer and the fault-tolerant remote terminals. What this means is that voting of sensor input signals and actuator command signals will have to take place at or near the site of the sensor and actuator sets. These redundancy management functions could be implemented as discrete devices, but they could be more practically and flexibly implemented as programmable functions of the fault-tolerant remote terminal components. As a result, the proposed components should now be called programmable fault-tolerant remote terminals or PFTRTs.

Fault Tolerance—The fault-tolerant nature of the proposed programmable fault-tolerant remote terminal (PFTRT) should consist of its ability to manage the large number of multiplexing, interfacing, and redundancy management functions required of it without losing control of the basic MIL-STD-1553A protocol. The

worst case fault that could occur at a PFTRT is for it to simultaneously babble on all of the bus lines to which it has access. This condition could effectively bring down the entire fault-tolerant bus network and result in a major FTFCS failure. Mechanisms for guarding against the occurrence of such an event at remote terminals currently exit, but not for a remote terminal of the complexity of the proposed PFTRT. This is an area of fault-tolerant design implementation that clearly requires more development.

A.2.3 FTFCS Packaging Considerations

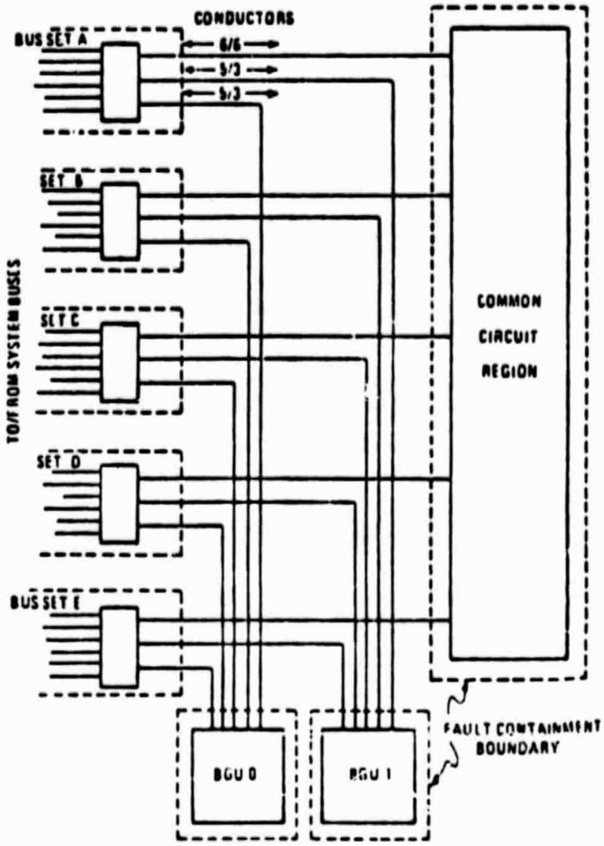
This section discusses the packaging alternatives that might be applicable to the implentation of realistic FTFCS designs. Special attention is given to the possible impact of dramatic advances in digital circuit technology.

Advances in the state of the art in digital hardware technology will, to a large extent, reveal themselves in FTFCS component packaging considerations. In particular, the advent of VLSI technology will allow almost unlimited architecture and packaging tradeoff possibilities. The number of packaging alternatives possible for a given FTFCS architecture is vastly reduced, however, when one takes into consideration the factors of electrical and mechanical design constraints, the functional nature of the FTFCS, the FTFCS aircraft environment, and the concept of line replaceable units (LRU).

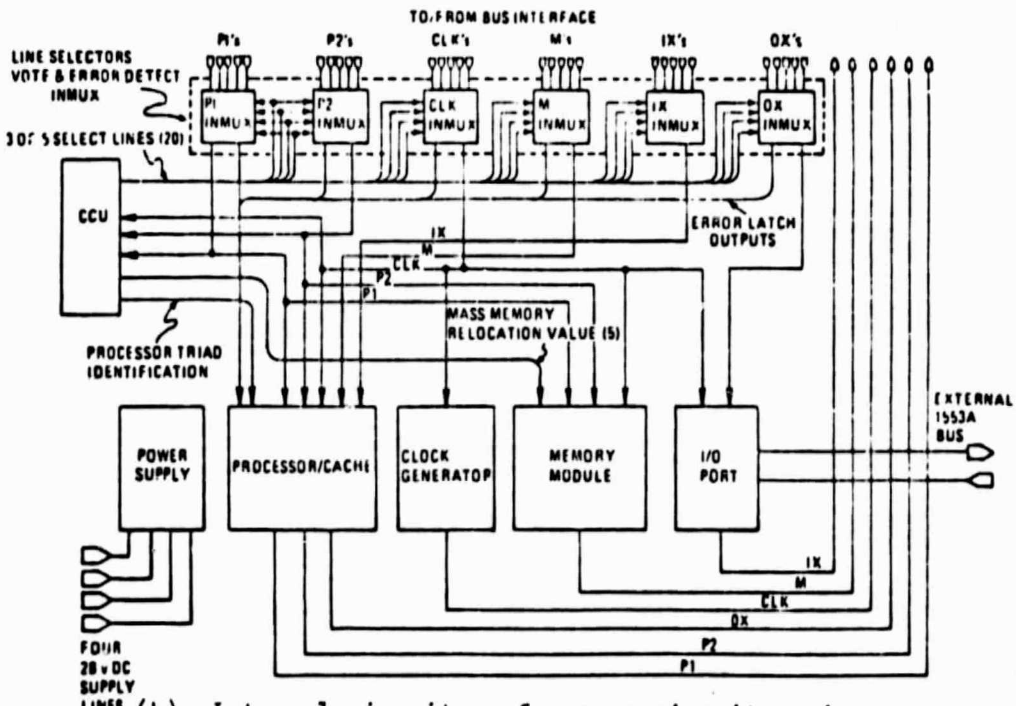
When the components of an FTFCS are examined from a high-level architectural point of view, many different packaging alternatives seemingly practical and desirable are, in fact, unrealistic. Consider, for example, the FTMP central computer. A user of the CBOM might propose an examination of the economic impact of packaging alternatives. FTMP contains a number of physical fault-containment regions that play an important role in the isolation of malfunctioning electronic components. If an FTMP LRU were to be broken up, it would appear to make sense to do so at the boundaries of the existing fault containment regions, not at the boundaries of the reconfigurable components (fig. A-8a). The complexity of the interconnection circuitry of the components found in the common circuitry fault containment region (fig. A-8b) provides an argument against packaging an FTMP with stage-based LRUs.

The realization of some component packaging alternatives could have detrimental effects on the concepts of particular FTFCS architectures. An example of this phenomenon is the separate packaging of a reconfigurable component, such as a digital I/O interface, that has a strong electronic dependency or association with some other major component, such as a central processor. In addition to disturbing the electrical nature of the original LRU that housed these components, such an alteration might profoundly change the basic complexity of the FTFCS involved. For instance, in the case of the separate packaging of the I/O interface, an entirely new component might have to be created to facilitate communication between the I/O interface component, now housed in its own LRU, and the central processor component, which is still located in the original LRU. If it should turn out that the

ORIGINAL PAGE IS
OF POOR QUALITY



(a) Fault containment boundaries



(b) Internal circuitry of common circuit region

Figure A-8. FTMP Fault-Containment Regions

new component must, for some reason, itself be reconfigurable, then the functional level of complexity of the FTFCS will have been increased. The penalty to be paid for this increased complexity will probably be a requirement for an upgrade of the FTFCS central computer's global executive and component reconfiguration control software.

The overall aircraft environment will also place restrictions on the array of packaging alternatives for a given FTFCS. The fact that the many physical devices, such as sensors, actuators, and computers that perform the functions of an FTFCS, are physically distributed about the aircraft makes certain packaging alternatives unrealistic. Some schemes are obviously unworkable; i.e., one would not want to propose that the FTFCS central computer be packaged at the site of a particular sensor set, say on a wing. Other packaging schemes may prove to be infeasible for reasons that are not as obvious. For example, it might be proposed that the remote terminals of a MIL-STD-1533A data bus be located at the site of the FTFCS central computer and be packaged as one remote terminal per LRU in a common equipment rack with data transfers between the sensor/actuator sets and the remote terminals taking place via a shipboard digital data bus system. A potential flaw in this organization is that it does not take into account that sensor/actuator redundancy management and analog data conditioning and conversion may be most efficiently implemented by the integration of the circuitry to perform these functions with the remote terminal circuitry to form smart fault-tolerant remote terminals; in that case, the optimal packaging arrangement would seem to be that of one smart remote terminal per LRU with the LRU located in close proximity to its associate sensor or actuator set to minimize shipboard wiring.

The rapidly advancing state of the art of digital technology will undoubtedly affect the concept of physically discernable digital components. A central processing unit (CPU) on a chip is a reality today and the realization of a computer on a chip appears close at hand. Indeed, the continued successful application of VLSI technology can only lead to the eventual possibility of an entire multiprocessing computer system, such as SIFT or FTMP, being implemented on a single integrated chip.

What is important when considering packaging alternatives that the CBOM should be capable of addressing is not what is possible but rather what is reasonable. Fundamental to the concept of an FTFCS is the inherently discrete physical nature of major reconfigurable system components and their packaging in LRUs. The advent of a SIFT or an FTMP implemented on a single chip would contradict this concept, so such an implementation could not be considered a reasonable possibility. An FTFCS central computer consisting of an arrangement of single-element fault-tolerant computers, say SIFTs, can be hypothesized, but such a system would most likely have the individual SIFT systems as reconfigurable components, again packaged in some configuration of LRUs and subject to many of the physical constraints of existing digital circuit implementations.

The conclusion drawn here is that while dramatic advances in the field of digital circuit integration will occur, the discipline of implementing and packaging these components in a commercial flight control system environment will in many respects not change. Components will still fail and need to be replaced and repaired and so will be most conveniently packaged as a collection of traditional

LRU types. Some major changes will, of course, take place. For example, the inclusion of integrated circuits in sensor and actuator component hardware and the corresponding elimination of special purpose device interfacing and signal conditioning components is a likely trend. Changes such as this introduction of smart sensors and actuators, however, does not change the important design concepts of an FTFCS and, in fact, enhances the view of an FTFCS architecture consisting of a fault-tolerant central computer communicating, via a fault-tolerant digital data bus, with various fault-tolerant digital control and measurement devices.

It is the rule rather than the exception that physical and functional design constraints, FTFCS aircraft environment considerations, and the concept of line replaceable units will eliminate many apparently useful economic alternatives of packaging FTFCS components. This is not to say that realistic packaging alternatives will not exist, but only that their numbers will be greatly reduced. This section concludes with a specific example of an FTFCS component for which some realistic packaging alternatives, other than that originally specified by the system designers, are possible. The component chosen is the SIFT I/O port, which, according to current design specifications, is an integral part of the standard SIFT LRU (sec. A.2.2.1). The alternative chosen for consideration is that the SIFT I/O ports be packaged as one or more separate LRUs. This alternative is examined in terms of its potential impact on the remaining circuitry of the original SIFT LRU, the functional nature of a SIFT processing system, the components other than those of the SIFT in the FTFCS, and the important concept of line replaceable units.

In the SIFT architecture, an I/O port is simply a MIL-STD-1553A bus controller (fig. A-9). Data entering and leaving the I/O port communicates with the SIFT processor/memory module via a special I/O buffer, rather than by gaining access to the processor's main data bus. A special interface is provided so that the SIFT processor can access this I/O buffer. As a result, packaging the SIFT I/O ports separately from the standard SIFT LRU would have a minimal impact on the remaining circuitry in SIFT LRUs, requiring only that a data bus be provided for the transfer of data between the new I/O port LRUs and the original I/O port buffers. The SIFT interprocessor broadcast bus is capable of providing each SIFT LRU with a copy of the data received by any one I/O port supplying any output data to any or all I/O ports so the SIFT I/O port can be considered to be a major reconfigurable component; thus, the separate packaging of the I/O ports does not disturb the functional view of SIFT and, in fact, may enhance this view. Data transmission considerations require that the new I/O port LRUs be placed in close proximity to the other SIFT LRUs (most reasonably in the same equipment rack); therefore, there will be no impact of this packaging alternative on the FTFCS components other than SIFT or on the FTFCS aircraft environment in general. Finally, separate packaging of SIFT I/O ports will neither detract from nor enhance the concept of line replaceable unit in an FTFCS.

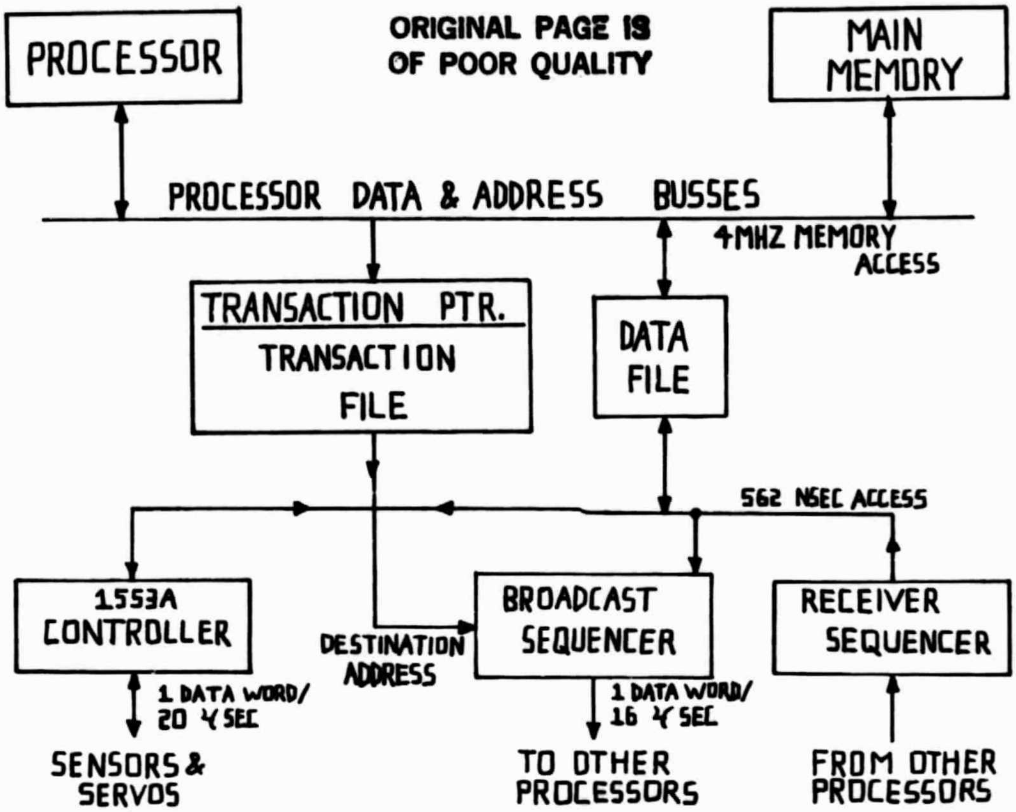


Figure A-9. SIFT Computer

A.3 SURVEY OF FTFCS FUNCTIONAL CHARACTERISTICS

This section discusses the functional characteristics of the various subsystems comprising several different FTFCS designs. Primary attention is given to the FTFCS central computers, especially SIFT and FTMP. External devices such as sensors, actuators, data busses, and remote terminals are also discussed but, as no fully integrated FTFCS architecture exists for these devices, the examined systems will be hypothetical.

Each FTFCS central computer and each hypothetical external device grouping will be examined according to the following functional characteristics:

- Application of tasks and scheduling
- Global executive
- Clock synchronization
- Broadcasting
- Error reporting
- Reconfiguration
- Voting

A.3.1 SIFT

An essential characteristic of SIFT is the ability to detect a fault in a processor module. A fault is detected by voting, and voting is performed on the outputs of applications of global executive tasks. Only malfunctions that cause a disparity among the voters will be detected.

With a risk of oversimplifying some important steps, how and when the voting takes place and the results will be discussed.

In SIFT, tasks are scheduled periodically according to the priority strategy shown in Figure A-10a. To illustrate voting, some details of the scheduling process will be explained. The highest priority frames (approximately 20 ms) are divided into subframes (about 2 ms) with each task assigned to a specific subframe depending on its voting dependencies. Prior to scheduling a task, the executive gathers the task's input data from producing processors, votes these data, and then releases them to the task about to be scheduled. Lower priority tasks are similarly voted but are not dependent upon their scheduling sequence within their priority frame. The lower priority tasks have their outputs double buffered and use as inputs data produced during the previous time frame. Figure A-10b shows this double buffering. Even with the high priority task scheduling, SIFT is designed to allow up to 50 s of skew between processors.

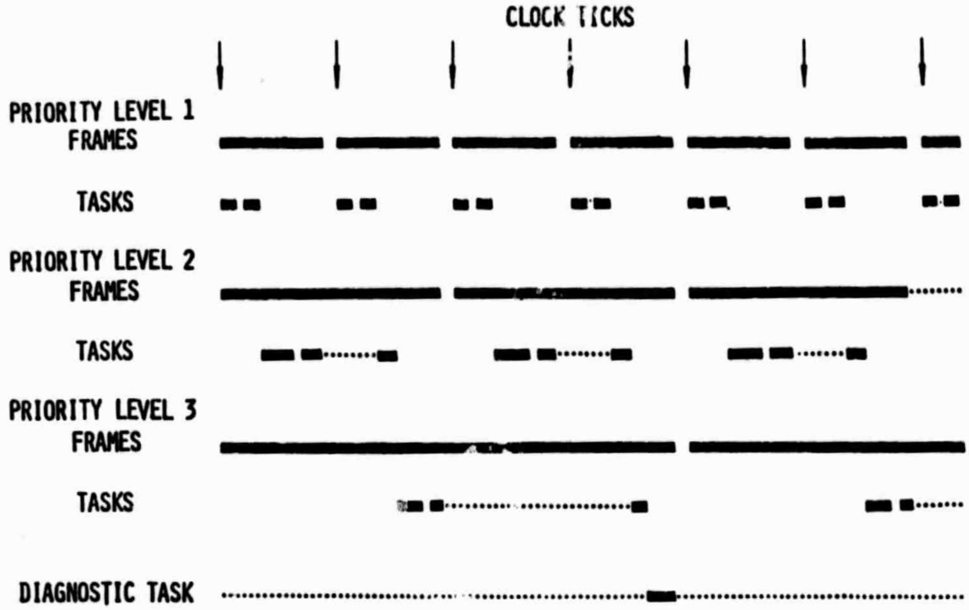
When an error is detected by voting, the error is masked and recorded in a processor error table. The offending processor, however, remains active until an error count threshold is reached at which time the processor is declared faulty and its tasks are reallocated, as shown in Figure A-10c. A brief algorithmic description of the voting and masking process follows:

- Look in the buffer area for each processor and do the following:
  - Look for buffer address of desired value
  - If buffer offset is active

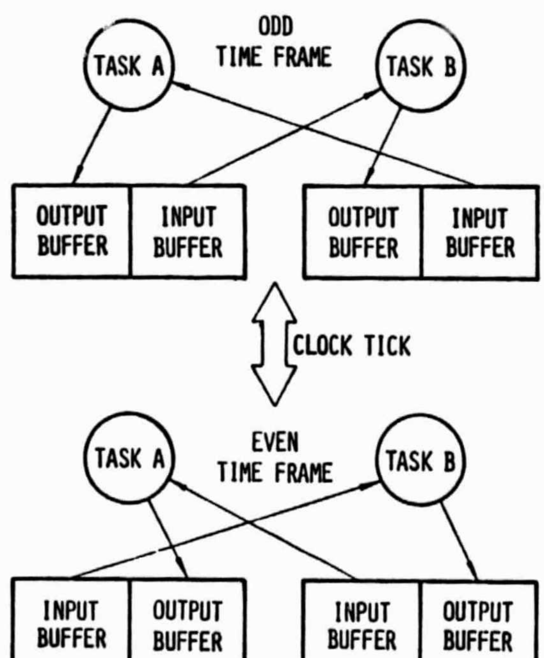
    Then assign buffer to set W
    Else assign buffer to set Z
  - Read value and check for consensus
  - If consensus exists

```
Then begin
    If value = consensus value
            Then assign buffer to set X
            Else assign buffer to set Y
    If buffer in set Y

            Then begin
                  Set buffer value to consensus value
                  Set flag in error table
                  End
    End
Else fill all buffer values in set W with SAFE value
```

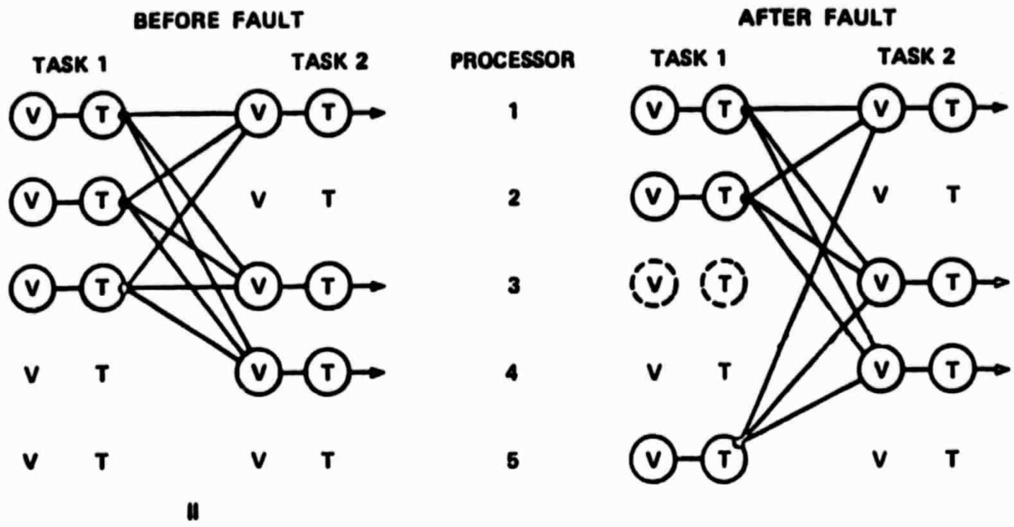


(a) Priority scheme



(b) Double buffering mechanism

Figure A-10. SIFT Functional Characteristics



(c) Reconfigurable voting

Figure A-10. SIFT Functional Characteristics (Concluded)

In a system of SIFT processors, no single processor has permanent or temporary hegemony. Each processor has its own local executive. A global executive also exists and is run as a triplicated periodic task. The local executive performs the following functions:

- Schedules tasks
- Votes input data and reports errors
- Handles task output buffers
- Handles errors locally

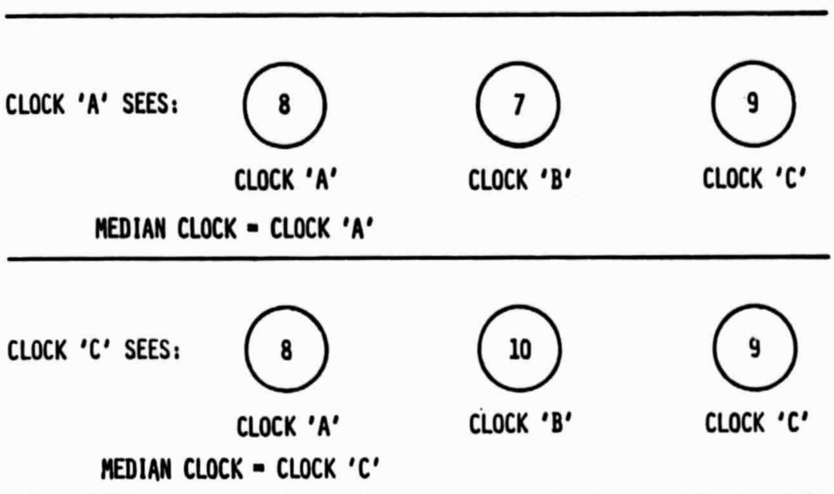
The global executive performs the following functions:

- Monitors error tables to look for processors with permanent faults
- Allocates tasks to processors
- Handles reconfigurations due to changes in flight phase

It can be seen from the above admittedly simplistic discussion of the SIFT fault-tolerant implementation that no hardware mechanisms are used to detect faults or to manage the system reconfiguration. Thus, the model definition of reconfigurable components turns out to be very simple. Essentially there is only one reconfigurable component—the processor. The processor is used for whatever tasks allocated to it by the global executive until a permanent fault is detected. In the event of a permanent fault, the processor's tasks are all allocated to other processors. The faulty processor is ignored by its fellow processors even though it may write information into their broadcast interface.

The SIFT design approach is not restricted to the BDX 930 computer system but could be used with other processors. The Bendix engineering test model, however, will serve as the basis for cost, reliability, power, and other physical characteristics of this component definition (refs. 4 and 6).

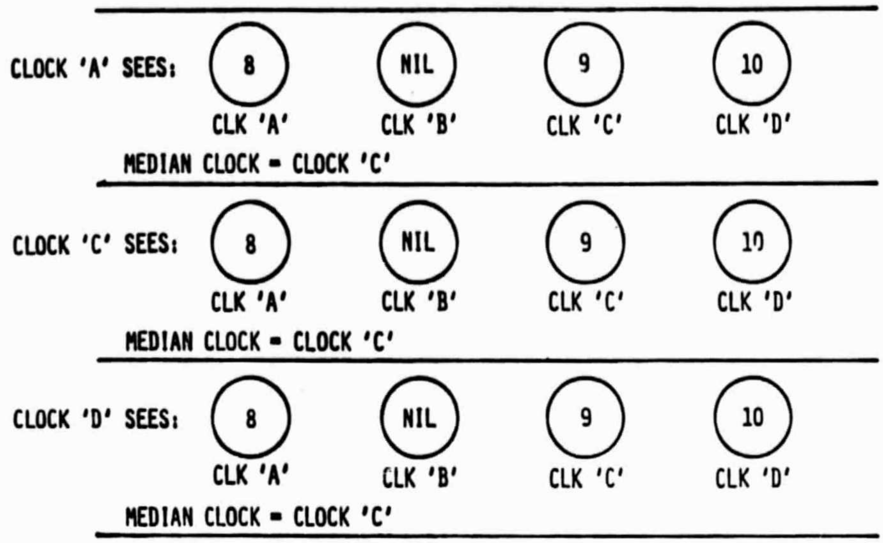
In a SIFT system, the amount of redundancy employed is dynamic and is a function of the criticality of a given task and the current state of the system. One implication is that, in the presence of several successive failures, a SIFT could be gracefully degraded in steps from a system of several TMR microprocessing channels to a single nonredundant channel. However, the clock generator synchronization algorithm employed (figs. A-11a and A-11b) requires that at least four SIFT LRUs with four nonfailed clocks be operational. Thus, the number of failed LRUs that can be tolerated in a five processor SIFT is one.



IN THE CASE WHERE A CLOCK FAILS SUCH THAT IT CAUSES TWO GOOD CLOCKS TO 'SEE' IT DIFFERENTLY, THE MEDIAN CLOCK ALGORITHM MAY FAIL AND THE GOOD CLOCKS MAY DIVERGE.

(a) Three clocks--one clock failed

Figure A-11. SIFT Clock Scheme (Continued)



HERE, EACH CLOCK TAKES A MAJORITY VOTE OF THE VALUES 'SEEN' FOR A GIVEN CLOCK BY ALL OTHER CLOCKS.

IF NO SUCH MAJORITY EXISTS, THEN A VALUE OF 'NIL' IS GIVEN TO THE PARTICULAR CLOCK. THUS, DIFFERING READINGS OF ONE FAILED CLOCK DO NOT DESTROY THE MEDIAN ALGORITHM.

(b) Four clocks--one clock failed

Figure A-11. SIFT Clock Scheme (Concluded)

A.3.2 FTMP

While the fault detection and masking of FTMP are implemented by hardware devices, system configuration, reconfiguration, and task assignment are software implemented.

The executive program is responsible for maintaining the state of the system. This includes initialization of the system into the following configuration:

- Three processor/cache channels—each a triad (9 of 10 processors used)
- Three main memory pages—each a triad. Page one contains system and applications programs and is written in nonvolatile form (ROM). Page two contains dynamic variables. Page three is not needed for fully configured operation (9 of 10 processors used).
- One clock quad (4 of 10 clocks used).
- Several I/O ports—one per active tasks.

Figure A-12a shows a fully configured system; note that the presence of a failed module within an LRU does not affect the ability of other modules in that LRU to be configured into triads or quads.

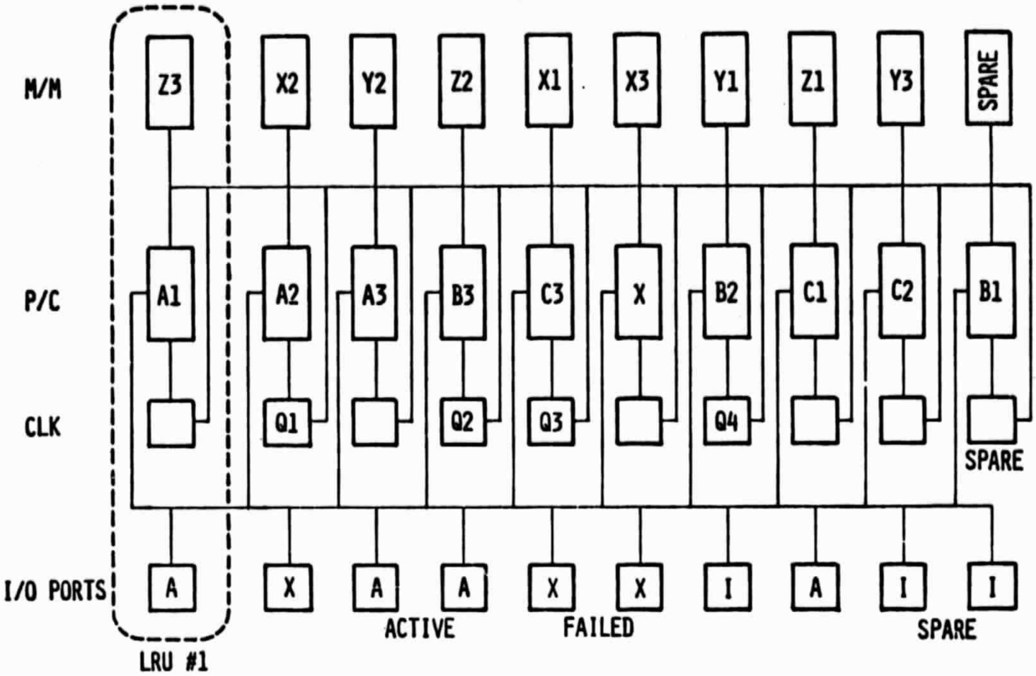
The module configuration explained above is accomplished by the assignment of the proper bus lines to active modules. A fully configured active bus network consists of the following configuration:

- One O-line triad. Each member of a P/C triad talks to main memory over one member of the current O-line triad (three of five lines used) (fig. A-12b).
- One I-line traid. Each member of a main memory page sends data to a P/C over one member of the current I-line triad (three of five lines used) (fig. A-12c).
- One C-line quad. Four clocks (in phase locked loop) are necessary to prevent drift; each module selects three out of four active C-lines to form its clock traid (four of fives line used) (fig. A-12d).

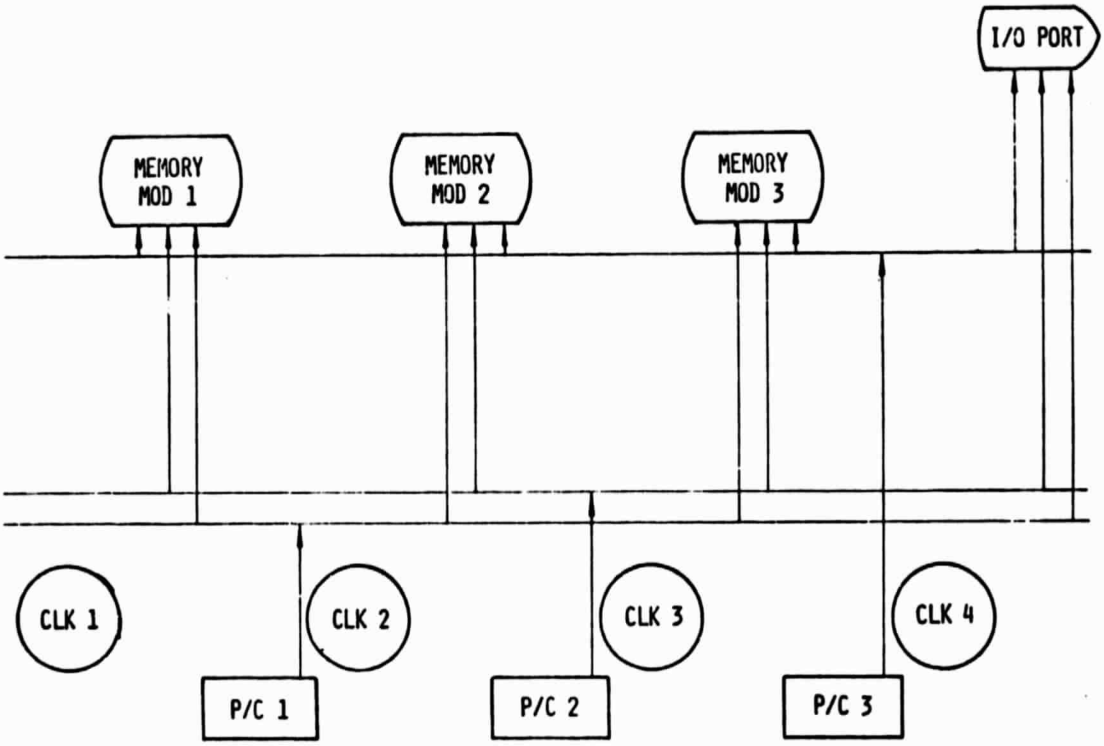
All five P-lines are available to poll the current bus triads and quad for access.

The bus assignments previously described (as determined by the executive) are maintained by a dual redundant bus controller, the bus guardian unit (BGU), in each LRU. Bus assignment consists of the controller's latching of the correct lines in the bus interface. The bus interface and the two BGUs are not reconfigurable and thus, must control all data flow to and from the components in their LRU, no matter how these components are configured throughout the system.

Tasks are scheduled from a common task pool. Upon successful completion of a task, a processor triad executes the next available task from the pool. This task will be run to completion, without interrupts, unless system self-testing or error-recovery routines require reconfiguration of the task's particular triad.



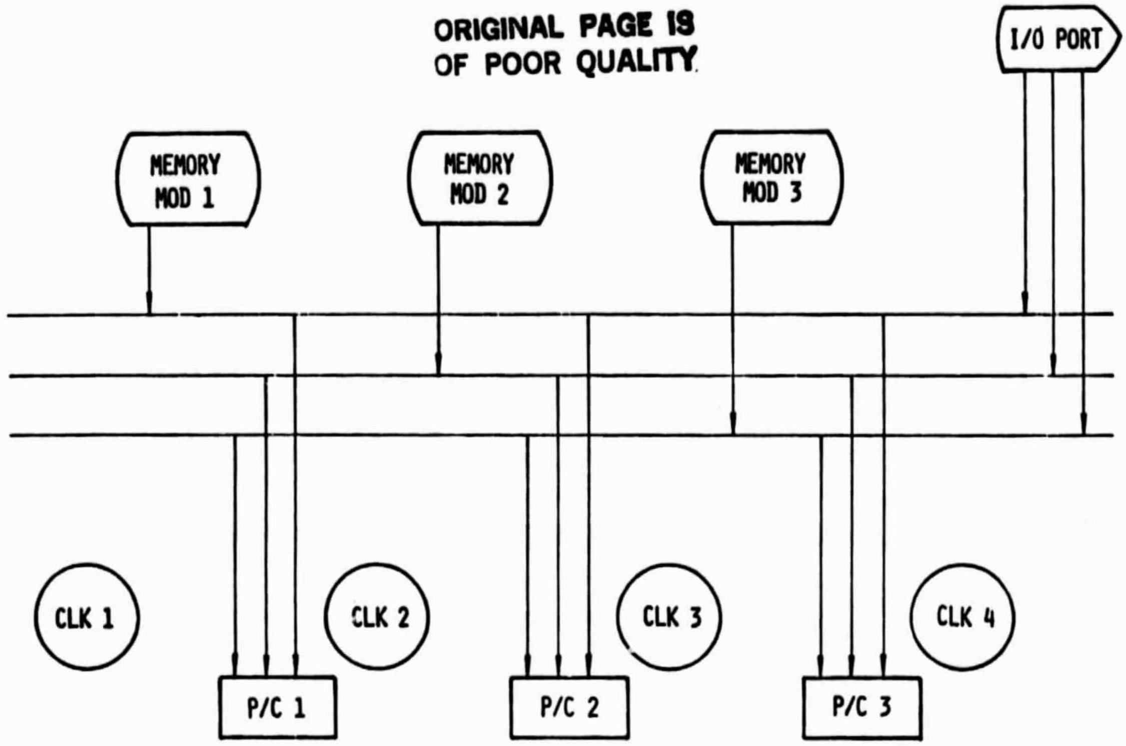
(a) Redundancy view



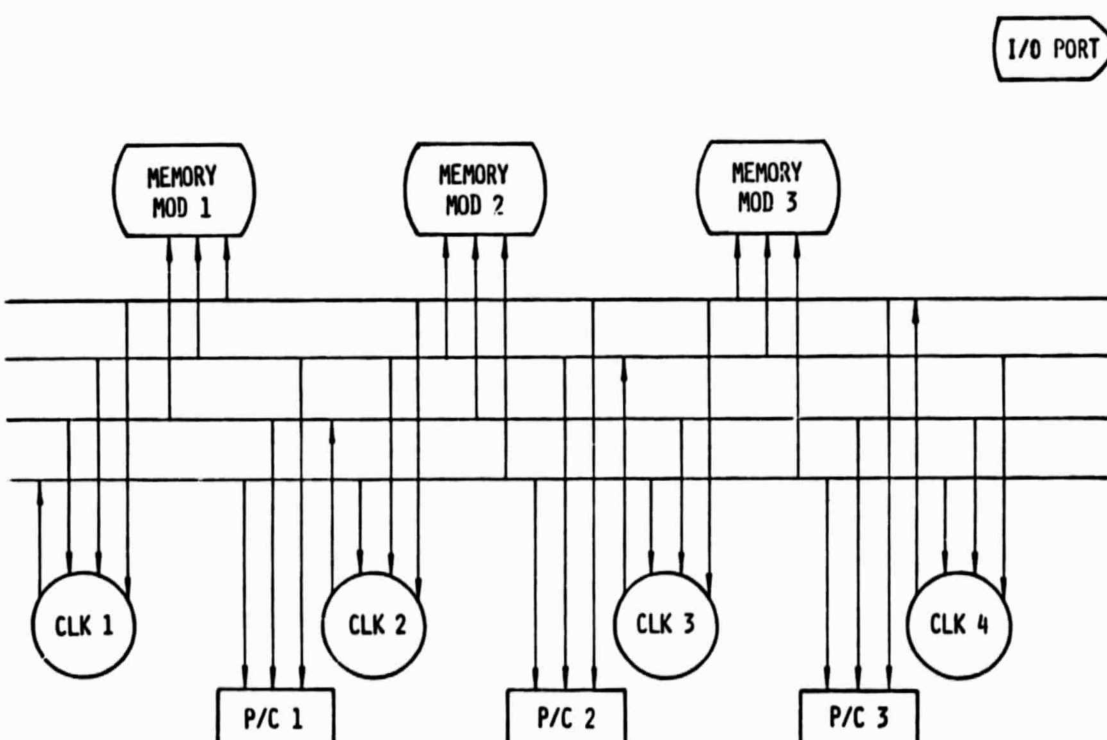
(b) Interconnection scheme (0-lines)

Figure A-12. FTMP Functional Characteristics

ORIGINAL PAGE IS
OF POOR QUALITY



(c) Interconnection scheme (I-lines)



(d) Interconnection scheme (C-lines)

Figure A-12. FTMP Functional Characteristics (Concluded)

Figure A-13 and the following FTMP reconfiguration algorithm demonstrate the steps the executive program initiates in recovering from a faulty processor-to-memory data transmission.

- Triad A sends data on O-lines, 1, 2, 3
- Voter detects data disagreement and sets error latches
- A processor triad notices error latch
- Error latch localizes error to O-line 1 or its associated processors
- Task list will localize error to P1 bus line 1
- Next memory write occurs
- Votor detects data disagreement and sets error latches
- Error latch localizes error to O-line 2 or its associated processors
- Previous error exposes (via error table) P1 as failed unit

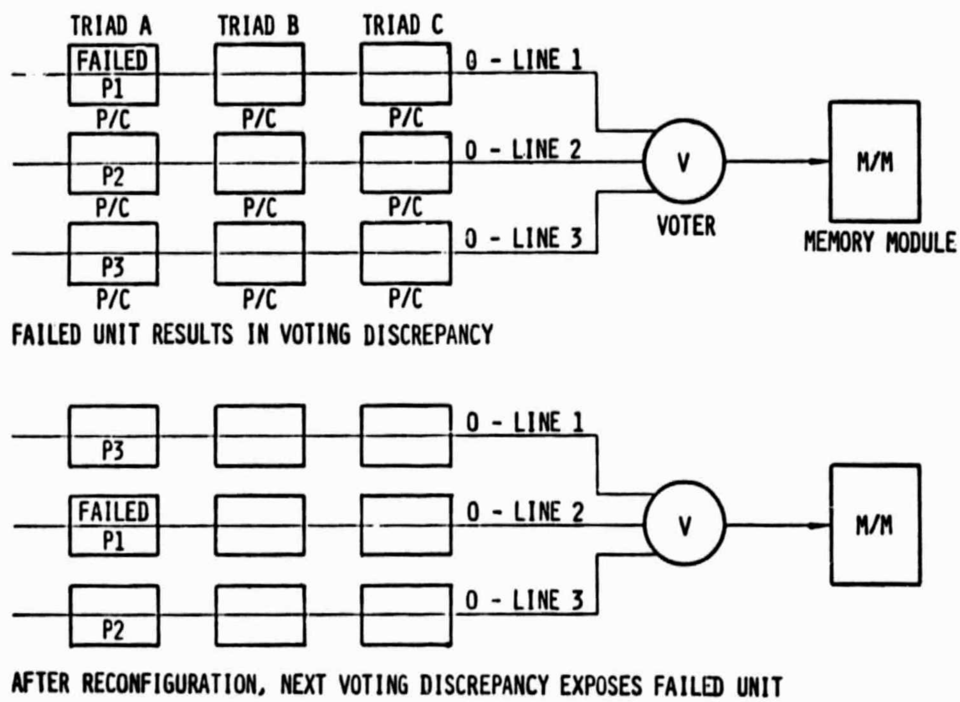


Figure A-13. FTMP Reconfiguration

In this example, processor number 1 of Triad A is failed and is the source of the erroneous data. The data disagreement is detected by the hardware voters associated with the destination main-memory modules (a triad). These voters will automatically set hardware error-latch registers, indicating which bus line the faulty data bit was transmitted on. The executive program periodically scans these registers and, if an error-latch is set, initiates reconfiguration. This reconfiguration consists of disassociating suspected data source modules from suspected data busses; further voting discrepancies will pinpoint the source of faulty data. In addition, an error table is kept that tabulates the number and rate of faults caused by each of the reconfigurable modules in the system. This table is used to determine when a unit should be considered failed and, therefore, be brought offline.

The executive is also responsible for graceful system degradation due to exhaustion of spares. In the case of processor/cache modules, this entails dismantling one triple-redundant processor channel and designating its members as spares. This, of course, cannot be done if there is only one active triad. In this case, degradation consists of operating as a dual-redundant channel; further degradation will be catastrophic.

There is no graceful degradation of clock generators; if less than four clock generators are available, synchronization cannot be guaranteed.

Because only one I/O port is necessary for system operation, up to nine failed ports can be tolerated. System performance will be degraded, however, in that active processors and sensors will be competing for access to the remaining ports.

Memory page degradation consists of the following steps:

- Since page 3 is not necessary for system operation, it may be dismantled and its members used as spares without degrading performance.
- If a member of the page 1 triad fails, it must be replaced with a volatile spare and thus, a degree of protection is lost.
- As page 1 and page 2 memory modules are not interchangeable, failure of more than four modules will result in dual redundant configuration of the affected page.

A.3.3 Fault-Tolerant Software

The subject of software reliability has not yet been fully addressed by the designers of the various fault-tolerant systems examined during the making of this report. In all of these systems it is required that the software executing on the control computers approach 100% reliability in order for the reliability of the complete FTFCS not to be degraded.

In the case of SIFT and FTMP (the only fault-tolerant central computers examined that are of sufficient reliability to be included in an FTFCS), the several years of development and testing of the executive software should result in these executive programs being highly reliable; however, this is not guaranteed. Further, the ultimate reliability of a given FTFCS will be dependent on the uncertain reliability of future application software.

In many respects the problem of software unreliability is analogous to that of hardware unreliability. Detectable and recoverable faults in components or unrecoverable system failures can be traced to either design (or manufacturing errors) or to the known property of life-cycle degradation of electronic or mechanical components (i.e., wearout characteristics). The former cause is very serious in its effects in that design or manufacturing errors will likely be shared among all components of a particular type and thus have the potential of creating unrecoverable fault situations. Physical component wearout is a far less serious occurrence; it is, in fact, the assumed cause of all component failures in a fault-tolerant system and provides the justification for combating the effects of detectable faults.

Many of the same techniques used to minimize hardware design and manufacturing errors can be used to minimize software design, coding, and installation errors. A maturing of the sciences of structured algorithm development, software quality control, and program module testing is required. An example of this approach is the proof-of-correctness approach taken by SRI in the development of the SIFT control computer's executive software. In fact, SRI has extended this concept so that the nature of the handling of applications tasks by the SIFT executive requires that these tasks be implemented and installed in small, precisely timed code segments, each of which can be thoroughly and independently tested. While this implementation structure is likely to be impractical to use in an engineering environment, it does represent progress in the control of inherent software errors.

Unlike electronic or mechanical components, dissimilar redundant software modules have no physical structure that can wear out. One can speculate that software modules can be expected to exhibit characteristics similar to those of their physical component counterparts. This fact should allow software modules to be treated by a user of the CBOM in the same manner as he treats hardware components. What is required is that the practices of controlled software development discussed in the previous section be followed in order to create libraries of software modules (components) that are as free of errors as possible. These software components, or larger software components built from the smaller basic software building blocks, then can be assigned failure rates based on their complexity (the number of basic software building blocks involved) and the apparent failure rates of the basic building block components used in their construction. The analogous process of assigning failure rates of electronic components based on the complexity of their circuits (e.g., chip counting) and the known failure rates of standard electronic parts (chips) should serve as a pattern for the development of this methodology.

It is expected that at least some of the application software that might be implemented on FTFCS control computers will involve the use of fault-tolerant software techniques and mechanisms. The following is a brief tabulation of several such techniques and mechanisms surveyed in making this report.

Deadline Mechanisms—This method requires the installation of an application-level scheduler that uses the system clock to note the passage of real-time intervals. Several versions of a particular task are supplied to the scheduler. The different versions each require a different amount of computational time for execution. The version requiring the longest execution time will do the most complete job of the task at hand. The next version will require somewhat less execution time but will not perform some desirable but nonessential portion of the task. This pattern continues, with versions requiring successively less computational time performing successively fewer functions of the given task; the final, shortest version will perform only those functions necessary to prevent system failure, perhaps merely supplying results of the previous invocation of its associated task as current data. The scheduler is supplied with a dealine time interval for each version of a given task, past which the next smaller version of the task would not have time to run to completion in the current iteration of the control system. The scheduler first invokes the longest version of the task and if at the expiration of the deadline time interval this version has not been completed, then the next largest version is invoked. Again, if this version has not been completed by the expiration of its deadline time interval, then its execution is aborted and the next largest version is

invoked. This process is continued until one version completes its execution within its allotted time interval or until the smallest version is invoked. It is, of course, necessary that the nature of the smallest version of a given task be such that there is near 100% certainty that it will always run to completion within its allotted deadline time interval. While this deadline mechanism approach does not, in and of itself, solve the problem of faults generated by miscalculation of data items, it does offer a safeguard against faults generated by timing problems of tasks whose length of completion may be dependent on the nature of input data or other unknown factors.

Self-Testing Software—One widely known technique in the area of fault-tolerant software is the development of algorithms that include self-testing functions. The most common form of self-testing software uses so called reasonableness-of-value parameters to determine the relative accuracy of intermediate or final data calculations. Under this method, a control task would occasionally compare calculated data items with those contained in a list of representative data values or with those computed during the previous control loop iteration. If the currently computed values differed from those contained in the reasonableness-of-value parameter list, then the task would report itself in error to the executive or some watchdog task. Corrective action could then take place such as task restart or rollback, deadline rescheduling, or the invoking of dissimilar software. The potential danger of self-testing software is the possibility of errors in the self-test algorithms themselves that may result in the failure to supply coverage of all software faults or even in the reporting of nonexistent faults.

Restart/Rollback Mechanisms—This technique requires the presence of some method of fault detection (self-test or external data monitor) and a connected executive or applications-level task rescheduler. The method used here is that once a software error is detected, a task (or set of tasks) is either restarted or rolled back to one of several rollback points that have been implemented at the time of algorithm development.

Dissimilar Software—The practice of simultaneously executing two or more independently developed dissimilar versions of the same task is intended to minimize the occurrence of faults caused by coding or logic errors. When all such versions of a given task have completed execution, their results can be compared or voted and majority values can be selected or discrepancies can be reported to an external fault monitor. This technique requires the implementation of software or hardware voters and/or discrepancy monitors.

A.4 FTFCS MODELING—CONCLUSION

The information presented here on FTFCS design practices represents the results of a survey of fault-tolerant mechanisms and techniques as they are, or might be, implemented in modern commercial transport aircraft flight control systems. As stated in the introduction, the specific goal of this effort was to assist in establishing the levels of abstraction and detail required when providing FTFCS descriptions as inputs to the CBOM and to give a potential user an idea of the range of systems the CBOM can model.

A.5 REFERENCES

1. Rice, J. W. and McCorkle, R. D., "Digital Flight Control Reliability—Effects of Redundancy Level, Architecture, and Redundancy Management Technique," AIAA 79-1893, 1979.

2. McCorkle, R. D., "Economically Reliable Digital Flight Control Using Actively Managed Redundancy," Boeing Aerospace paper presented at the Naval Postgraduate School, Monterey, California, July 11-13, 1978.

3. Wakerly, J. F., "Microcomputer Reliability Improvement Using Triple Modular Redundancy," IEEE proceedings V66, No. 10, October 1978.

4. Wensley, J. H. et al., "SIFT: Design and Analysis of a Fault-Tolerant Computer for Aircraft Control," IEEE proceedings V66, No. 10, October 1978.

5. Hopkins, A. L. et al., "FTMP: A Highly Reliable Fault-Tolerant Multiprocessor for Aircraft," IEEE proceedings V66, No. 10, October 1978.

6. Weinstock, C. B., "SIFT: System Design and Implementations," IEEE proceedings 1604-8, 1980.

APPENDIX B

EXPECTED DISPATCH DELAY COSTS DUE TO FAILURES OF AN FTFCS STAGE

This appendix provides additional detail on the mathematical derivation of the dispatch delay formulas presented in Section 4.2.

Assumption

The stage in question contains n redundant units of which m are assumed to be on line at the start of a maintenance cycle, and the remaining n-m units are assummed to be working and either "hot," "warm," or "cold" standbys. Two maintenance policies are evaluated for each stage. These policies correspond respectively to maintenance policies 3 and 1 described in Section 4.2.

- Policy A: restoration to new condition (n) every T flight hours (scheduled maintenance) or at stage dispatch failure (unscheduled maintenance), whichever comes first

- Policy B: restoration to new condition (n) every T flight hours (scheduled maintenance) and partial restoration to dispatch minimums (m) at stage dispatch failure (minimal repair)

The basic quantity of interest is the expected number of nondispatch incidents (ND) (or stage failures) calculated as follows:

For policy A

$$ND = E\,(N_Z(T)) \cdot \frac{F}{T}$$

where

F	=	total flight hours/year
T	=	flight hours per maintenance cycle
$N_Z(t)$	=	the random number of stage failures in the interval (0,t), given n units operational at time 0.

For policy B

$$ND = A(T) \cdot m \cdot \frac{F}{T}$$

where A(T) = the expected time duration within a maintenance cycle of length T of the "barely operational" state; i.e., no standby units available, and λ is the failure rate of the unit.

The following assumptions are made for units within a stage:

- Online units fail independently with exponential time to failure of rate λ

ORIGINAL PAGE IS
OF POOR QUALITY

- Standby units fail independently and exponentially with rate $\theta\lambda$ per unit time, $0 \leq \theta \leq 1$. $\theta = 1$ is a hot standby, $\theta = 0$ is a cold standby.
- No failures occur during nonflight times; when an online unit fails, switchover to a standby is perfect and instantaneous.
- The stage fails when fewer than m units are operational.

Exact analytic expressions for ND have been obtained for each of the above policies under these assumptions. Direct numerical evaluation of these expressions is possible. The basic tools for deriving the analytic results are the Markov properties of the exponential distribution and renewal theory.

Analytic expressions for these quantities make use of the random variables Y and Z defined by

$$Y = X_n + X_{n-1} + \cdots + X_{m+1}$$

$$Z = X_n + X_{n-1} + \cdots + X_m$$

where X_j is the time to failure of the i^{th} component, assumed to be an exponential r.v. with rate

$$\lambda_j = \left[m\lambda + (j-m)\theta\lambda\right] = \begin{cases} j\lambda \text{ for hot standby} \\ m\lambda \text{ for cold standby} \end{cases}$$

$\theta = 0$ is cold standby; $\theta = 1$ is hot standby; $0 < \theta < 1$ is a warm standby.

Z is the time to stage failure from a completely restored condition, and Y is the time to stage barely operational from a completely restored condition (used to derive A(T) for policy 2).

Exact analytic expressions for the probability density functions $f_Z(t)$ and $f_Y(t)$ as sums and differences of exponential functions have been developed. The relationships of A(T) in policy 2 to $f_Y(t)$, and E(U) in policy 3 to $f_Z(t)$, are fairly straightforward, so that exact expressions for these quantities are available. Computationally, these analytic expressions (even though exact) may present a problem: there may be a tendency for adjacent terms in the polynomial expressions to nearly cancel out, leading to numerical instability. Until this issue can be resolved, it is preferable to stay with Gamma approximations to $f_Y(t)$ and $f_Z(t)$ (discussed below). We take each of the policies in turn:

ORIGINAL PAGE IS
OF POOR QUALITY

For policy A

$$ND = E(N_z(T)) \cdot \frac{F}{T}$$

and the problem is to compute the renewal function $E(N_z(T))$. One expression for this function is

$$E(N_z(T)) = \sum_{j=1}^{\infty} \text{Prob}\left[N_z(T) \geq j\right]$$

$$= \sum_{j=1}^{\infty} \text{Prob}\left[S_{zj} \leq T\right]$$

where

S_{zj} = Time to the j^{th} stage failure, a sum of j independent r.v. with the Z distribution.

Therefore,

$$\text{Prob}\left[S_{Zj} \leq T\right] = F_Z^{*j}(T),$$

where F_Z^{*j} is the j-fold convolution of the $f_Z(t)$ distribution function. The exact distribution of a Z random variable can be obtained by looking at the Laplace transform $m_z(s)$ of its density function:

$$m_z(s) = E(e^{-sz}) = \int_0^{\infty} e^{-st} dF_z(t) = \int_0^{\infty} e^{-st} f_z(t)\, dt$$

because

$$Z = \sum_{j=m}^{n} X_j$$

F_Z is the convolution of $r' = n-m+1$ distinct exponential r.v.'s with rates λ_j. The transform $m_z(s)$ is the product of the $m_{x_j}(s)$, which are

$$m_{x_j}(s) = \int_0^{\infty} e^{-st} \lambda_j e^{-\lambda_j t}\, dt$$

$$= \frac{\lambda_j}{s+\lambda_j}$$

ORIGINAL PAGE IS
OF POOR QUALITY

So that

$$m_z(s) = \prod_{j=m}^{n} \left(\frac{\lambda_j}{s+\lambda_j}\right).$$

This product can be decomposed by partial fractions, leading to

$$m_z(s) = \sum_{j=m}^{n} \frac{a_j}{s+\lambda_j},$$

with a_j determined as follows:

(a) Equate the two expression for $m_z(s)$:

$$\sum_{j=m}^{n} \frac{a_j}{(s+\lambda_j)} = \prod_{j=m}^{n} \frac{\lambda_j}{s+\lambda_j}$$

(b) Clear denominators:

$$\sum_{j=m}^{n} \left[a_j \sum_{\substack{k=m \\ k\neq j}}^{n} (s+\lambda_k) \right] = \prod_{j=m}^{n} \lambda_j.$$

(c) Successively substitute $s = -\lambda_m, -\lambda_{m+1}, \ldots, -\lambda_n$ on the LHS of this expression:

$$a_j \prod_{\substack{k=m \\ k\neq j}}^{n} (\lambda_k - \lambda_j) = \prod_{k=m}^{n} \lambda_k, \; j = m, m+1 \ldots, n$$

or

$$a_j = \frac{\prod_{k=m}^{n} \lambda_k}{\prod_{\substack{k=m, \\ k\neq j}}^{n} (\lambda_k - \lambda_j)} \qquad \text{for } j = m, m+1, \ldots, n.$$

now substituting $\lambda_k = m \quad + (k-m)\theta\lambda$ and redefining indices of multiplication, (new k = old $k-m$).

$$a_j = \frac{\lambda^{n-m+1} \prod_{k=0}^{n-m} (m+k\theta)}{(\theta\lambda)^{n-m} \prod_{\substack{k=m \\ k\neq j}}^{n} (k-j)} = \frac{\theta\lambda \prod_{k=0}^{n-m} \left(k + \frac{m}{\theta}\right)}{\prod_{\substack{k=m \\ k\neq j}}^{m} (k-j)}$$

and

$$m_z(s) = \prod_{j=m}^{n} \left(\frac{m\lambda}{s+m\lambda}\right) = \left(\frac{m\lambda}{s+m\lambda}\right)^{n-m+1}$$

which is the Laplace Transform of a Gamma distribution. So the partial fraction decomposition is valid for $0 < \theta \leq 1$.

The product in the denominator of a_j can be further analyzed, and one gets

$$a_j = \frac{(-1)^{j-m}\, \theta\lambda \prod_{k=0}^{n-m} \left(\frac{m}{\theta} + k\right)}{(n-j)!\, (j-m)!}$$

Note the alternating signs. Now, since $\frac{1}{s+\lambda_j}$ is the Laplace transform of the exponential $e^{-\lambda_j t}$, the density function

$$f_z(t) = \sum_{j=m}^{n} a_j e^{-\lambda_j t}$$

$$= \sum_{j=m}^{n} a_j e^{-(m+(j-m)\theta)\lambda t} = e^{-m\lambda t} \sum_{k=0}^{n-m} a_{m+k} e^{-k\theta\lambda t}$$

If this expression is used to generate the convolutions $F_Z^{*j}(T)$ needed to calculate the renewal function, the result will be exceedingly complex. Therefore, $f_Z(t)$ will be approximated by a Gamma density of the form:

$$f_Z(t) \approx g_Z(t) = \beta e^{-\beta t} \frac{(\beta t)^r}{r!} \sim \text{Gamma}(\beta, r+1)$$

where $r = n-m$, in which case $F_Z^{*j}(T) \approx G_Z^{*j}(T)$ and G_Z^{*j} is Gamma $(\beta, j(r+1))$.

Since $G_Z^{*j}(T_m) = \int_0^T \beta e^{-\beta t} \frac{(\beta t)^{jr+r-1}}{(jr+r-1)!} dt$

ORIGINAL PAGE IS OF POOR QUALITY

$$= 1 - e^{-\beta T} \sum_{k=0}^{jr+r-1} \frac{(\beta T)^k}{k!},$$

one expression for the renewal function is

$$(1) \quad E(N_Z(T)) \approx \sum_{j=1}^{\infty} \left[1 - e^{-\beta T} \sum_{k=1}^{(j+1)r} \frac{(\beta T)^{k-1}}{(k-1)!} \right]$$

An alternative is to start from

$$E(N_Z(T)) = \sum_{j=1}^{\infty} j \left[F^{*j}(T) - F^{*(j+1)}(T) \right],$$

using the Gamma approximation to get

$$(2) \quad E(N_Z(T)) \approx e^{-\beta T} \sum_{j=1}^{\infty} j \left[\sum_{k=(j+1)r+1}^{(j+2)r} \frac{(\beta T)^{k-1}}{(k-1)!} \right].$$

For computational purposes (2) may be better than (1) in the range $\beta Tr < 10$, but some testing is in order. The first M terms of the infinite series should be summed, where M is chosen so that the relative error due to truncating the series is less than some preselected value (say 10^{-5}). The value of M may be preset if testing reveals that it is not larger than 3 or 4 for all $\beta Tr < 10$. Otherwise it will be necessary to sum a variable number of terms, determining M at the time of computation within the subroutine that calculates $E(N_Z(T))$.

Two choices of β for the Gamma approximation suggest themselves. One is the ordinary mean

$$\beta = \frac{1}{(n-m+1)} \sum_{j=m}^{n} \lambda_j = m\lambda + \frac{(n-m)}{2} \theta\lambda$$

ORIGINAL PAGE IS OF POOR QUALITY

$= m\lambda$ if $\theta = 0$ (cold standbys - Gamma distribution exact)

$= \left(\frac{n+m}{2}\right)\lambda$ if $\theta = 1$ (hot standbys)

The other is the harmonic mean

$$\beta_h^{-1} = \frac{1}{(n-m+1)} \sum_{j=m}^{n} \frac{1}{\lambda_j}$$

which has the property that

$$\int_0^{\infty} t g_z(t)dt = E(Z) = \int_0^{\infty} t f_z(t)dt$$

So that the mean time to failure for the Gamma approximation agrees with the exact value. It is not clear that this unbiasedness property is of any practical importance.

For Policy B

The derivation of the exact distribution for Y is almost identical to that for Z. The Laplace transform of the density function is

$$m_Y(s) = \prod_{j=m+1}^{n} \frac{\lambda_j}{s+\lambda_j}$$

The partial fraction decomposition now has coefficients

$$b_j = \frac{(-1)^{j-(m+1)}\theta\lambda \prod_{k=1}^{n-m}\left(\frac{m}{\theta} + k\right)}{(n-j)!\ (j-(m+1))!}$$

for $j=m+1, m+2, \ldots, n$ and

$$f_Y(t) = \sum_{j=m+1}^{n} b_j e^{-\lambda_j t} = e^{-m\lambda t} \sum_{k=1}^{n-m} b_{m+k} e^{-k\theta\lambda t}$$

The Gamma approximation is now

$$g_Y(t) = \beta e^{-\beta t} \frac{(\beta t)^{r-1}}{(r-1)!} \sim \text{Gamma } (\beta, r)$$

and the value of β is

$$\beta = \frac{1}{(n-m)} \sum_{j=m+1}^{n} \lambda_j = m\lambda + \frac{(n-m+1)}{2} \theta\lambda$$

or

$$\beta_h^{-1} = \frac{1}{(n-m)} \sum_{j=m+1}^{n} \lambda_j^{-1}.$$

APPENDIX C

MODEL ENHANCEMENTS

Several suggestions are made in Appendix C for model enhancement. These are not essential to the model, but they could extend its usefulness. Seven enhancements are presented:

- Failure rate data base
- Stage availability optimization
- Markov model
- Dispatch reliability formulas
- Repair shop capacity
- Maintenance interval optimization
- Dynamic inventory optimization

With the possible exception of the failure rate data base, each of these enhancements represents a small additional programming requirement, and each should be implemented. Equations and algorithms that might be used are presented in this appendix.

Failure Rate Data Base

A useful enhancement might be a data base of component failure rate data for the types of equipment in the design. To the extent possible this could draw on existing data systems and incorporate guidelines for relating part count to MTBF for different categories.

It would be particularly valuable to have reasonable ways to determine past experience or to predict the following:

- MTBF
- Removal verification rate
- LRU condemnation rate
- Average repair costs

It might be possible for the designer to describe the type of equipment and allow the computer to predict the MTBF as well as computing the effect on the system. Further research would be required to determine the feasibility of this enhancement.

Stage Availability Optimization

This enhancement is to calculate the proportion of time a noncritical system or stage is available. For nondispatch critical systems, the consequences of a functional failure may be so minor that the repair is considered deferrable, and the aircraft continues flying without use of that particular function. Such systems do not necessarily have immediate emergency repairs and they do not cause dispatch delays. Therefore, cost optimization cannot be based on dispatch delays as it can be with dispatch-critical FTFCS systems.

In these cases the stage availability enhancement allows the designer to do the cost optimization. Instead of dispatch delays, the basic tradeoff is with availability, or the proportion of flight hours in which the system is available. This is an important enhancement for many systems for which dispatch delay is not an appropriate measure.

Stage availability is calculated for a given maintenance policy as a function of spares coverage P. In the policy of deferring maintenance until the next overnight stop where a replacement spare is available, the following is the solution:

Let

F	=	annual flight hours for fleet
L	=	average daily flight hours of an aircraft
SF	=	the number of times that a stage fails
P	=	spares coverage: the probability for any aircraft that the next scheduled loading is at an airport with an LRU in stock
D	=	the expected duration in flight hours that a failed stage remains unavailable

D can be computed by observing that whenever a stage failure occurs, the stage is restored that day with probability P, for expected duration L/2. It is restored on the second day with probability (1-P)P, for expected duration of 3L/2 and so on. This is a convergent series that has the following solution:

$$D = \frac{L}{2} P + \frac{3L}{2}(1-P)P + \frac{5L}{2}(1-P)^2 P + \ldots$$

$$= \frac{L}{2} + L \, \frac{1-P}{P}$$

ORIGINAL PAGE IS
OF POOR QUALITY

and availability (A) may thus be computed:

$$A = \frac{F - SF \times D}{F} = \frac{F - (SF) \times \left[\frac{L}{2} + L\left(\frac{1-P}{P}\right)\right]}{F}$$

Because the stage failures (SF) may be computed by the same mathematics used for nondispatch incidents (ND) in the basic model, this equation completes all that is necessary for calculating availability.

In order to optimize nondispatch critical systems, a penalty is assigned for unavailability (much as it was for dispatch delays). A reasonable default value may be computed by the following logic: suppose a system has a cost of ownership of 1000 units, but it is only available 70% of the time it is needed. Allocating the cost to the percentage of time the system is available yields $\frac{1000}{70} = 14$ monetary units per percentage point of availability. This may be used as a default value for the unavailability penalty and the resulting system optimization should be reasonable. The analysis will determine the sensitivity of this parameter.

Stage availability is a useful, feasible enhancement of the optimization model, especially for analysis of non-dispatch-critical systems. It also is useful for analyzing dispatch-critical systems that are integrated with noncritical functions. In such a case, the enhanced model can weight the unavailability of the noncritical functions.

ORIGINAL PAGE IS
OF POOR QUALITY

Markov Model

The basic optimization model uses a stage analysis for computing dispatch delay rates, by computing delay rates for each individual stage and then summing them. This approach gives valid results in most cases of analysis; however, there are at least two possible cases when it would be desirable to have a more complex model. The first case is if cost estimates are needed for systems with intrinsically high delay rates. The basic method would overestimate such rates because it counts the event of two dispatch failures on the same flight as two separate delays instead of one. The amount of overcounting is the product of the delay rates from the two stages. The second case is if an FTFCS proves to be overly difficult to partition into a component incidence matrix. This could be the case for systems designed with unusually complex logic for equipment utilization.

For these occasions the nondispatch incidents may be estimated by solving a Markov model. This model is similar to that used in the reliability design modeling, except for three important differences:

- The model includes repairs as well as failures.
- The model is not nearly as precise as that required for reliability (and therefore it is much more efficient).
- The model does not include simultaneous failures or coverage, but is limited to equipment levels.

Many proven solution methods are available for this enhancement. It requires little more than an appropriate matrix multiplication program. This enhancement is also useful for making preliminary reliability estimates for screening the feasibility of certain decision variables in the optimization without the necessity of a detailed reliability analysis.

Dispatch Reliability Formulas

The following enhancement allows the analyst to estimate the inherent dispatch reliability of a stage. The inherent dispatch reliability of a stage i is defined in terms of the length of time until the stage drops below dispatch conditions, assuming no interim maintenance.

The reliability function is calculated by the binomial probability equations for the length of time that it takes for n-m+1 failures to occur.

The solution is:

$$R(T) = n! \sum_{i=1}^{n-m+1} \frac{e^{-\lambda(m-1+i)}}{(m-1+i)!(n-m+1-i)!} (1-e^{-\lambda T})^{(n-m+1-i)}$$

ORIGINAL PAGE IS
OF POOR QUALITY

The mean time between stage failures (again, no interim repair) is obtained from the following formula:

$$MTSF = \frac{1}{\lambda} \sum_{i=1}^{n-m+1} \frac{1}{(n+1-i)}$$

This information may be used to determine upper and lower bounds on the component removal rate, independent of the maintenance policy by the following logic. The maximum removal rate occurs if every failed component is immediately replaced. In this case there are always n components failing and the maximum component failure rate for the stage is $n\lambda$.

On the other hand, if components are not replaced at all until the stage becomes nondispatchable, then the corresponding average component failure rate is a reasonable lower bound. In this case there are n-m+1 components to be restored every MTSF hours for a component failure rate for the stage of

$$\frac{n-m+1}{MTSF}.$$

(The absolute minimum is realized if the stage is kept at level m, for rate $m\lambda$, but this is trivial because if it were optimal then necessarily n=m.)

Let $\overline{MTCR}$ = mean time between component removals (upper bound)

$\underline{MTCR}$ = mean time between component removals (lower bound)

Then

$$\overline{MTCR} = \frac{MTSF}{n-m+1}$$

$$\underline{MTCR} = \frac{1}{n\lambda}$$

These formulas provide a useful enhancement to allow the analyst to do some quick order-of-magnitude estimates without doing the whole optimization. Also these can be incorporated into the cost equations to give upper and lower bounds on the cost, which would occasionally be useful, especially for diagnostics.

Repair Shop Capacity

From the FTFCS design point of view, there are only two parameters of the LRU repair shop which have any importance: the average cost of repair and the average time in the repair shop. It makes no difference whether or not the repair shop is owned by the airline, the manufacturer, or someone else. It makes no difference to the designer how the repair shop organizes its internal processing, priorities, or schedules. All the designer needs to estimate is the average cost and the average time.

But from the repair shop's point of view, the situation is very different. The characteristic of a repair shop is that it does not have a regularly scheduled workload. The smaller the shop the more important this becomes. If an airline does not have a sufficiently large repair volume to smooth out the workload, then the repair shop must absorb excessive fluctuations, resulting in higher average repair costs or longer repair cycle times.

A detailed internal modeling of a repair shop is a complex queuing problem that often has no feasible solution other than by simulation. The basic relationships may be reasonably approximated, however, and used to develop guidelines for estimating the probable relationships between:

- Average demand on the repair shop
- Average direct cost of repair
- Average time an LRU is in the repair shop

Analysis shows that for planning purposes it is the average backlog in the repair shop that determines the feasibility of a repair shop, and the optimum average backlog depends on the cost of repair and the cost of capital. Furthermore, the average cycle time in the repair shop can be calculated from the same variables. Following are the basic formulas:

Let

C = average annual cost of repair shop, to be minimized

$$C = C_A + C_B + C_C$$

where

C_A = the direct cost of repairing LRUs (annual service hours)

C_B = the overhead cost of idle capacity (annual hours when there is insufficient workload)

C_C = the holding cost of idle equipment.

Assume that r and a are given, where

r = average direct cost of repair per LRU

a = average annual holding cost per LRU

Let

L = average numer of LRUs in the repair shop

LM = average annual repair shop demand

Then

ORIGINAL PAGE IS
OF POOR QUALITY

$C_A = r * LM$, and

$C_C = a * L$.

The capacity utilization (ρ) of a repair shop is the average proportion of time in which at least one LRU is in the shop. Assuming a constant service capacity of the repair shop, the utilization is:

$$\rho = \frac{C_A}{C_A + C_B}$$

or

$$C_B = C_A\left(\frac{1-\rho}{\rho}\right)$$

thus,

$$C(\min) = C_A + C_A\left(\frac{1-\rho}{\rho}\right) + a\,L$$

or

$$C(\min) = \frac{C_A}{\rho} + a\,L$$

The average number of LRUs in the (M/M/1) repair shop is a function of capacity utilization:

$$L = \frac{\rho}{1-\rho}$$

Thus,

$$C = \frac{C_A}{\rho} + a\left(\frac{\rho}{1-\rho}\right)$$

This equation is minimized with respect to ρ by taking the derivative and setting equal to zero:

$$0 = \frac{C'(\rho)}{\rho^2} = \frac{-C_A}{\rho^2} + a\,\frac{(1-\rho) + \rho}{(1-\rho)^2} = -\frac{C_A}{\rho^2} + \frac{a}{(1-\rho)^2}$$

or

$$0 = -C_A(1-\rho)^2 + a\rho^2$$

or

ORIGINAL PAGE IS OF POOR QUALITY

$$\frac{C_A}{a} = \frac{\rho^2}{(1-\rho)^2} = L^2$$

Therefore the optimum backlog in the shop is

$$L = \frac{\rho}{1-\rho} = \sqrt{\frac{C_A}{a}} = \sqrt{\frac{rLM}{a}}$$

and the optimal average repair time (W) is therefore

$$W = \frac{L}{LM} = \sqrt{\frac{r}{a*LM}} \quad \text{(years)}$$

For example, if LM = 360 LRUs per year, a = \$4000 (annual holding cost of an average LRU for a year) and r = \$200 average direct repair cost of an LRU, then the optimal backlog in the shop (L) is

$$L = \sqrt{\frac{200 \times 360}{4000}} = 4.4 \text{ LRUs}$$

and the optimal average repair cycle time is

$$W = \frac{4.4 \times 365}{360} = 4.5 \text{ days}$$

The above approach is a feasible enhancement to the model for calculating a reasonable default value for the repair cycle time.

In order to test the sensitivity of the assumptions used in the derivation of the equation for repair time, W, the optimum average backlog of LRUs in a repair shop, was calculated for three different repair shop models:

- (M/M/1): single server, random arrival, exponential service time
- (M/K/1): single server, random arrival, constant service time
- (M/M/2): two server, random arrival, exponential service time

The corresponding equations for the optimal L in these three repair shop models turn out to be the following:

(M/M/1): $L = \rho/(1-\rho)$

(M/K/1): $L = (2*\rho-\rho^2)/(2-2*\rho)$

(M/M/2): $L = 2*\rho/(1-\rho^2)$

The following table was the result of the test, suggesting that the (M/M/1) model is not an unreasonable approximation to a complex repair shop even though it is derived from an unsophisticated model. Furthermore, the proposed (M/M/1) model for obtaining default estimates of repair shop delay appears reasonable because the

average time spent in the repair shop delay appears reasonable because the average time spent in the repair shop is a relatively small portion of the repair cycle time. The larger portion of repair cycle time is due to routine administrative delays, shipping time, and nonworking time caused by weekends or partial shifts.

REPAIR SHOP UTILIZATION	OPTIMUM BACKLOG IN REPAIR SHOP (L)		
	(M/M/1)	(M/K/1)	(M/M/2)
0.76	3.2	2.0	3.6
0.80	4.0	2.4	4.4
0.84	5.3	3.0	5.7
0.88	7.3	4.1	7.8
0.92	11.5	6.2	12.0
0.96	24.0	12.5	24.5

Maintenance Interval Optimization

The maintenance interval optimization enhancement provides the analyst with a tool that is capable of optimizing the maintenance period for a variety of policies. The method is developed in this appendix for a single stage. No specific method has been developed for the more realistic problem of optimizing two or more stages simultaneously. With appropriate airline cost data, however, the maintenance interval for two or more stages can be optimized together with no additional complexity.

Let

i = the number of accumulated component failures in a stage since the last restoration

Assume that the stage is restored to level n every T flight hours. The restoration requires some expense for scheduling the time and place and doing the inspection.

Let

C_1 = cost of a scheduled inspection

When inspection reveals a faulty unit, the unit must be removed and replaced (R&R), at the same direct cost no matter when it occurs.

Let

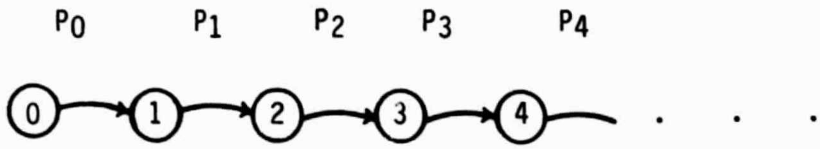
C_2 = the cost of a (nonemergency) remove and replace

If the stage is allowed to degrade to below dispatch minimum, an emergency repair must be made at a greater cost, the most important cost addition being due to the possibility of a dispatch delay/cancellation penalty.

Let

C_3 = the cost of an emergency (nonscheduled) remove and replace.

Next a Markov chain is defined for i = O, 1, 2, ... state space



by letting P_i = the probability that a stage in state i at time t will be in state i+1 at time t+1.

The state probabilities $\pi_i(T)$ are then computed by the following recursive formulas:

$$\pi_i(O) = \begin{cases} 1, & i = 0 \\ 0, & i = 1, 2, 3 \end{cases}$$

$$\pi_i(t+1) = P_{i-1} \times \pi_{i-1}(t) + (1-P_i) \times \pi_i(t)$$

The cost per maintenance cycle is computed by observing that because of the unusual definition of the state (as accumulated failures, whether or not repaired), each state at time T completely determines the number of scheduled remove and replace operations and the number of emergency remove and replace operations that took place in the interval. Accordingly the cost equation may be written directly as a function of T. For example, in the case where all nonemergency demand is deferred until the scheduled period:

$$C(T) = C_1 + \sum_{i=1}^{\infty} C_2 \times i \times \pi_i(T) + \sum_{j=n-m+1}^{\infty} C_3 \times (j-n+m) \times \pi_j(T)$$

This is the cost per cycle, and accordingly the minimum average cost is obtained by minimizing the single variable function:

$$T^*: \underset{\{T\}}{\mathrm{Min}} \frac{C(T)}{T}$$

Details are provided in Appendix E. This enhancement would be useful for investigating the sensitivity of the maintenance period. One of the limitations of this enhancement is the difficulty of obtaining data for C_1, C_2, and C_3. However, reasonable estimates can be made. This model is particularly valuable for analyzing stages that have more than 1 or 2 levels of redundancy above the dispatch minimum. For such stages the optimal maintenance policy is not necessarily condition monitoring, and this enhancement would provide a useful way to analyze the effect of the maintenance interval on the average cost.

Marginal Inventory Optimization

The basic optimization model uses a heuristic technique for determining the optimal inventory levels, namely by optimizing the number of line stations that are to stock spares and then treating the inventory level at any one station as a constraint problem. Every line station to carry inventory is constrained to have a minimum spares protection level (default value, 0.97). It is possible to develop more accurate approaches to the inventory optimization. The following enhancement has not been proven to be optimal, but it seems like a reasonable approach to improve the inventory optimization. The enhancement is based on the dynamic programming algorithm developed by Kettelle (ref. 1).

The basic idea of this enhancement is to marginally build up the stock from zero, each iteration adding one additional LRU to the location (either the depot or one of the line stations) that yields the largest marginal improvement in spares coverage P. If the cost function satisfies certain specific mathematical properties (convexity), then once an LRU is allocated to a given location it will always remain so allocated. In such a case, a valid stopping rule can be derived, and the total number of calculations required for FTFCS optimization can be reduced, perhaps significantly. The marginal allocation approach has been successfully applied to many similar types of spares provisioning problems, though not yet to the case where the number of line stations is a decision variable as well as the inventory levels. An unanswered question from the present project is whether or not the marginal allocation method will always yield an optimum. The question may be answered analytically (by proof or counter example) or empirically (by testing it against the heuristic in the basic mathematical model).

Using the formulas of Section 4.3, the allocation logic is the following. Allocate one LRU to the main base and compute P(1) = spares coverage provided by the optimal distribution of an inventory level of one spare unit. The second LRU added to inventory must go either to the depot warehouse, to the first base, or to the second base. Each of these alternatives can be evaluated to determine the maximum spares coverage, yielding P(2) plus the optimum distribution of two LRUs. The next LRU can be allocated in the same fashion. The addition of an LRU to the inventory by this method would never redistribute the preceding LRU distribution. Thus, the method is "marginal"; i.e., it allocates a inventory one step at a time. The marginal allocation method can be proven optimal under certain mathematical conditions, but it has not been determined whether or not these conditions apply to FTFCS optimization for a commercial airline.

If the inventory cost function is convex, the marginal inventory allocation enhancement can be expected to give more refined estimates of inventory requirements and to require substantially less computation than the basic method.

REFERENCES

1. Kettelle, John D., Jr., "Least-Cost Allocations of Reliability Investment," *Operations Research*, v. 10, 1962.

APPENDIX D

COMPUTER REQUIREMENTS FOR OPTIMIZATION

The following is an analysis of the computer requirements needed to implement the to optimization algorithms.

The analysis will consider the following topics in the light of an interactive package:

- Input
- Output
- Complexity of the algorithm
- Hardware requirements
- Options

Input

There are two kinds of input. The default (or background) input listed on page 22 and the foreground input that will be different from session to session.

The default input will reside in a file that will be read into the program by an initialization routine.

The foreground input may be divided into two types:

- LRU information
- Component information

Because the information abut LRUs is essentially constant, it should be structured in a data base so the analyst can concentrate on component specification. The data base should permit the selection of LRUs that meet multiple criteria; i.e., can perform some function and weigh less than X pounds. The size of the LRU data base may eliminate some computer systems from consideration; however, even desktop computers can provide a fairly rapid access to a data base containing a large number of LRUs.

The components would be specified in relation to the LRUs using the query language of the data base. This leaves only two items of data to be keyed in by the analyst: the component MTBF and the dispatch minimum complement. The necessary input requirements are therefore minimal and suitable for interactive sessions.

A useful feature of the input phase could be the option of storing defined components for later retrieval or changes, in effect creating a data base of components.

Output

Normally the analysis and output would be in a batch mode, especially if sensitivity analysis is desired. In the case of the optimization program, there is essentially only one datum that may be used as a basis to rank a given design. That is the cost of ownership. Designs with approximately equal costs may then be evaluated on the basis of additional information given by the model.

Complexity of the Algorithm

Sections 4.2, 4.3, and 4.4 of this report contain the basic equations of the algorithm. To approximate run time of the algorithm, the number of floating point operations required by these equations may be estimated. Because the optimization proceeds by going through several do-loops, unraveling their structure will give the total number of floating point operations required for the optimization of a stage.

The unit of measurement will be one multiplication, called the numeric cycle (nc). The other operations will be counted as follows: division = 2 nc's, exponentiation = 5 nc's, and addition = 1 nc. These values are typical of most processors, with the exception of addition, which is usually somewhat less than 1 nc. The purpose, though, is to look at the worst case behavior.

Next is a sketch of the essence of the optimization program outline. The notation used is:

(o)r	–	(optimal) redundancy level
(o)m	–	(optimal) maintenance policy
(o)co	–	(optimal) cost of ownership
P*	–	set of spares coverage factors
cf	–	spares coverage factor

```
for r = 0 to 2 do
  for m = 1 to 2 do
    for cf in P* do
      begin
      compute dispatch delays;
      compute spares provisioning;
      co = cost of ownership;
      update optimal variables;
end.
```

After the exit from these loops, the variable r will contain the optimal redundancy level, the variable m the optimal maintenance policy, and the associated optimal cost.

Next, consider the complexity of computing the dispatch delays, ND. The relevant equation is in Section 4.2.3 and it is estimated that it will take at most 20 nc's.

The situation with spares provisioning is more complicated. First it is necessary to compute the average resupply time $T(S_o)$ (sec. 4.3), which takes about $7 + 10\ S_o$. Next is computed the number of stations, J, which will stock parts, which may take at most 1/2 C(P), where C(P) is the size of the set of coverage factors (P). Finally the number of spares is minimized. The computation of $F(S_o, S_j)$ by equation 4-9 will take at least 10 nc's and will have to be done for every potential S_o and S_j. This contributes about $10 \cdot C(P) \cdot S$ computations. Altogether we have about $7 + 10 \cdot S + 10 \cdot C(P) \cdot S$, say

$$10 \cdot C(P) \cdot S$$

numerical cycles.

The cost of ownership calculation for EVCO in Section 4.4.1 is straightforward, taking about 25 nc's. Updating the optimal variables takes almost nothing.

When the above computations are put together, it may be estimated that the algorithm will require about

$$6 \cdot C(P) \cdot (45 + 10 \cdot C(P) \cdot S)$$

numerical cycles for each stage, which is optimized in the FTFCS.

Now the default value of C(P) may be 20, and S may be about 20 as well. The above figure then comes to slightly more than 480 000 nc's, or approximately 1/2 million. The worst case might occur with C(P) = S = 30, which would amount to about 1 620 000 nc's.

If C(P) is approximately the same magnitude as S, then the algorithm is cubic in C(P). The greatest improvements in performance can be found in the innermost loop, that is, in the minimization procedure described by equation 4-11. Appendix C describes an inventory allocation enhancement that may significantly reduce the number of these calculations.

Hardware Requirements

Having some specific numbers of operations that have to be performed, it is possible to specify how fast a processor should be for interactive work. Processors with known floating point operations rates (flops) are used to determine how long it would take to perform the estimated number of operations. This estimation technique does not account for the time taken by the control sequence of the code. Indeed, the only way to time a code is to run it. The figures developed, however, will be good enough to separate the feasible from the unfeasible candidate.

Consider six representatives of the computer spectrum and their performance in number of floating point operations per second. The last two columns in the following listing give the time (in seconds) to perform the optimization of a stage for C(P) = 20 (480 000 nc's) and C(P) = 30 (1 620 000 nc's).

		SECONDS	
COMPUTER	FLOPS	P = 20	P = 30
Programmable calculator	16	30 000	101 250
8-bit micro	500	960	3 240
16-bit micro	40 000	12	40
Minicomputer	100 000	5	17
Mainframe	1 300 000	0.4	1.4
Supercomputer	10 000 000	0.05	0.2

These are estimates. The table shows that on a programmable calculator the time to optimize each stage would take a good part of a day (perhaps days); on an 8-bit micro, a good part of an hour; on a 16-bit micro, a good part of a minute (perhaps minutes); and on a minicomputer, it might be under a minute.

For interactive processing it is important to have a response time in seconds. For this reason the FTFCS optimization, as it stands now, should be hosted on at least a minicomputer, although a good 16-bit microcomputer (with hardware floating point operations) might also give an adequate performance.

Options

The enhancements mentioned in the appendixes (other than marginal inventory allocation) would, of course, add to the time taken. This is true of the maintenance interval optimization. The algorithm for maintenance interval optimization was actually implemented on a minicomputer (VAX under UNIX), however, and for all reasonable ranges of the time period, the added time would be at most a few seconds.

The only option that might cause concern with performance is sensitivity analysis. The selected eight variables for sensitivity analysis (sec. 4.7) imply that all times above should be multiplied by eight, increasing the run times by almost an order of magnitude. A careful design may reduce the full sensitivity analysis run time by a factor of two or more by localizing the perturbation effects of the parameters. That is, when a parameter is perturbed its effect on some modules of the code is easily computable from the results of its effect on other modules. Identifying these dependencies and taking them into account in the code construction could produce some performance savings.

Another useful sensitivity analysis is in fact already built into the optimization approach. Specifically, it is the dependence of the life-cycle cost on the decision variables X (redundancy and the maintenance policy in the simplified version) and on the coverage factor P. The cost of ownership, EVCO, could be represented by a three-dimensional bar graph to make the driving factors immediately evident. The only additional requirement would be a simple graphics monitor. The graphic representation would have the additional benefit of enabling the analyst to readily discern that some redundancy level or coverage factors are, in a specific class of designs, always economically dominated by other choices. They could, therefore, be eliminated from consideration to improve the performance of the optimization algorithms.

APPENDIX E

OPTIMAL MAINTENANCE PERIOD

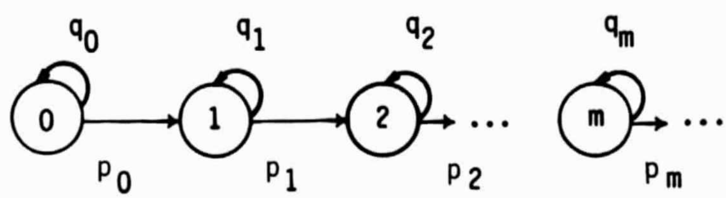
This appendix provides additional details of algorithms for maintenance interval optimization presented as an enhancement in Appendix C.

General Descriptions

Assume that a system consists of n components that fail at rate LM (per period of time); the system is operational if at least k components function. When the system is unavailable for service, we have the option of replacing (or repairing) all the failed components (full repair at line) or only the minimal number needed to make the system operational (minimal repair at line). Whenever the system fails, we incur a cost that is usually much larger than periodic renewal of the system. The purpose of the following algorithms is to find the optimal period of renewal for different policies of maintenance.

The Model

The relevant behavior of the system is described by a Markov chain



where

m = cumulative number of parts that have failed

p_i = the probability of transition from state i to state i+1

q_i = $(1-p_i)$

Let s_i^t denote the probability that the system has accumulated i failures by time t. We refer to s^t as the state vector at time t.

The first task is to compute the state vector for any t. As is often the case in Markov processes, the algorithm for doing so is recursive. The relations for elemental changes of the state vector are:

Forward:

$$s_0^+ = q_0 s_0$$

$$s_i^+ = q_i s_i + p_{i-1} s_{i-1}^+ \qquad (i > 0)$$

ORIGINAL PAGE IS
OF POOR QUALITY

Backward:

$$s_0^- = s_0/q_0$$

$$s_i^- = \frac{1}{q_i}\,(s_i - p_{i-1}\, s_{i-1}^-) \qquad (i>0)$$

where s_i^+ (s_i^-) denotes the probability of the system being in state i in the next (previous) transition.

The justification: when $i>0$, the transition into state i can come either from state i, which happens with probability q_i or from state i-1, which happens with probability p_{i-1}. Obviously, $s_0^+ = q_0 s_0$.

The backward relations follow from algebraic manipulations of the forward ones. They are needed for efficiency of searching.

The state vector will initially be set to (1,0,0,...) and the probabilities of transition (related to failure rates) will be generally quite small (<0.001). It follows from the relations given above that even after long periods of time the s_i^t will be small for sufficiently large i.

Our next task is to obtain the size δ (cumulative number of failed units) of the state vector that will give an approximation to the infinite vector with a specified degree of accuracy. More precisely, given that we want to find a state size δ so that if we make $p_d = 0$ and $q_d = 1$ (i.e., create an absorbing state at d), then $s_d^t < \delta$ for all user specified t, the restoration interval.

Let

$$(q_0, q_1, \ldots, q_m)^n = \sum q_0^{i_0} q_i^{i_1} \cdots\cdots q_m^{i_m}$$

Where the sum extends over all $(i_0, i_1, \ldots, i_m)$ so that

$$i_0 + i_1 + \ldots + i_m = n.$$

Note that there are $\binom{n+m}{m}$ of these summands.

The following claim, which gives a closed form expression for s_i^t can be checked by induction using the recursive relations for s_i^t.

$$s_i^t = p_0 p_1 \cdots p_{i-1}\,(q_0, q_1, \ldots, q_t)^{t-i}$$

with $t \geq i$).

Let p_{max}, q_{max} be maxima of p_i's and q_i's respectively (not considering $q_d = 1$). Then,

$$s_i^t \le p_{max}^i \cdot q_{max}^{t-i} \binom{t}{i}$$

In order to determine d, we increase i until the expression on the right-hand side becomes $<\delta$.

The computation of the expression can be achieved either by Stirling's approximation or logarithmic evaluation. More to the point though is to note that if the p_i's are small, $p_{max} \approx (1-q_{max})$ and the expression then becomes the Bernoulli probability of i successes in t trials, which in this case may be approximated by the Poisson distribution. Experimental evidence suggests that, with a proper choice of the Poisson parameter, this becomes a good approximation to s_i^t itself. In other words:

$$p_n(m) = \binom{n}{m} p^m q^{n-m} \longrightarrow \frac{(np)^m}{m!} e^{-np}$$

Algorithm Efficiency

Once the size of the state vector is fixed, say d, we can look at the number of operations needed to compute s^t. Counting only multiplications, their number is $2 \cdot d \cdot t$. Since t will generally be in the hundreds or thousands and will vary, we indeed have to set d as low as possible to increase the speed of the computation.

It should be noted that the computation of s (in either direction) can be done in place. In the forward direction we have computed s_i from d down while in the backward direction from o up.

The Cost Function

The cost function depends on the policy chosen but in either case it may be easily computed from a cost vector of the same length as the state vector.

Let

c_i = cost of inspection

c_r = repair of a component

c_d = cost of dispatch delay

For minimal repair at the line station the cost vector for i failures is:

$$c(i) = c_i + i \cdot c_r \qquad \text{if } i \le n - k$$

$$c(i) = c_i + i \cdot c_r + (i + k - n) \cdot c_d \quad \text{otherwise}$$

For full repair at the line station the cost vector per restoration cycle is:

$$c(i) = c_i + i \cdot c_r \qquad \text{for } i \le n - k + 1$$

$$c(i) = c_i + i \cdot c_r + b \cdot c_d \text{ otherwise,}$$

where b is the count of numbers divisible by n - k + 1 which are i.

Let $c = (c(o), c(1) \ldots c(d))$ be the cost vector (for either of the policies).

The function we wish to optimize is the average cost of the policy per period of time; i.e.,

$$K(t) = \frac{s^t * c}{t}$$

where $s^t * c$ is the scalar product of the state vector at time t and of the cost vector.

With properly chosen state size d, this function is unimodal in the range from which d was obtained.

This helps enormously for designing an optimization routine for a recursive function. Note that when we compute K(t) we have essentially computed K(t') for every $t' \le t$, since in order to compute s^t we have to compute $s^{t'}$ for every $t' \le t$.

The obvious algorithm for optimizing K is thus given by:

```
while K(t) > K(t+1) do
begin
    K(t): = K(t+1)
    t: = t + 1
    compute K(t)
end
```


The minimum of K is then at $(t - 1)$.

Depending on the range of t and the expected optimum we can almost halve the computation time by the following method:

- Precompute transition probabilities of going from i to i + 2.
- Go through the while-loop.
- Use backward relations to find the true optimum.

In general, the transition probabilities can be precomputed for steps of size a. After the loop it is known that the minimum lies between $j \cdot a$ and $(j+1)a$ for some j and the state vector at $(j+1)a$. Using steps of size 1 (or if t is large, a/2) the backward relations are used to get the true optimum.

The optimum is often of interest as well as in the interval (a,b), where the cost function differs from its optimal value by some factor !: if v is the optimal we want as large an interval (a,b) as possible such that for $t \in (a,b)$:

$$|v-K(t)| < \alpha \cdot K(t).$$

The case of a unimodal function can proceed as follows: let t^* be the optimal period and s the corresponding state vector.

To find b, compute (using the forward relations) K(t) for values t^*, $t^* + 1$, ... until K(t) exceeds $K(t) + \alpha K(t)$. To find a compute (using the backward relations) for values $t^* - 1$, $t^* - 2$, ..., until K(t) exceeds $K(t) + \alpha K(t)$.

To summarize, the routine for optimizing the maintenance interval has the following structure:

Input:		Output:
c_i, c_r, c_d	- cost coefficients	t - the optimal period
LM	- rate of failures	c - the corresponding cost
d	- size of state vector	a,b - tolerance (α) interval
mode	- which model	
α	- tolerance parameter	

ORIGINAL PAGE IS
OF POOR QUALITY